UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

INTEGRACIÓN ENTRE EQUIPOS DE NAVEGACIÓN SATELITAL Y PLATAFORMA DE GESTIÓN DE FLOTA

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN TECNOLOGÍAS DE LA INFORMACIÓN

MAURICIO LEONARDO HERNÁN VENEGAS MORALES

PROFESOR GUÍA:
JOSÉ A. PINO URTUBIA

MIEMBROS DE LA COMISIÓN:
FRANCISCO GUTIERREZ FIGUEROA
LUIS MATEU BRULE
PEDRO ROSSEL CID

SANTIAGO DE CHILE
2018

Resumen

Este trabajo se desarrolla bajo el contexto de estudiar la incorporación de equipamiento de navegación Garmin nüvi a la plataforma de gestión de flota de la empresa Citymovil S.A. y así proponer una solución que permita integrar y expandir las funcionalidades globales del sistema.

La problemática que se quiere abordar es apoyar el proceso de gestión de visitas a clientes, en empresas que realicen principalmente operaciones en terreno. El cumplir eficientemente con un itinerario se hace complejo, debido a la incertidumbre en los tiempos de traslado y atención. Además, la comunicación entre despachadores y conductores puede generar desviaciones por diversos motivos. El incumplimiento recurrente de los horarios de visita programados genera al corto plazo una mala percepción de la empresa con sus clientes.

La solución propuesta se basa en la construcción e implementación de varios módulos de software que faciliten la comunicación del itinerario de visitas entre los tres roles de usuarios identificados para este problema: Despachador, Conductor y Administrador. Técnicamente, el núcleo del desarrollo está enfocado en aprovechar las funcionalidades de gestión de georreferencias de equipos de navegación, que se utilizan como interfaz visual a bordo de los vehículos, lo que implica implementar instrucciones del protocolo propietario del fabricante.

Parte importante del esfuerzo dedicado está en el análisis y diseño de la integración de los componentes del sistema, siendo un aspecto muy útil para esta tarea el trabajar sobre una arquitectura de micro servicios.

La evaluación de la solución se basa en medir los niveles de satisfacción de los usuarios antes y después de la realización de una prueba piloto. Estos usuarios pertenecen al área de soporte y a técnicos en terreno de la compañía.

Luego de finalizar este trabajo, se puede concluir que el objetivo general de construir un sistema que integre equipamiento a bordo de vehículos, se logró de forma efectiva. No obstante, desde el punto de vista de los objetivos secundarios, es decir, el incremento en la satisfacción de los usuarios del sistema, el cumplimiento fue parcial. Esto se puede explicar debido al bajo nivel de madurez del área, lo que dificulta el sacarle un mayor provecho a intentos de incorporar nuevas tecnologías en el proceso de despacho. Queda como trabajo posterior realizar un estudio más específico y ver cómo mejorar el proceso de gestión de las visitas a terreno en el área de soporte.

Agradecimientos

A mi familia y amigos por darme el ánimo y motivación necesarios para abordar este desafío.

Un especial agradecimiento al profesor José A. Pino por la paciencia y su buena disposición en dirigirme durante el desarrollo de este trabajo.

A los profesores miembros de la comisión: Francisco Gutierrez, Luis Mateu y Pedro Rossel por sus comentarios en favor de mejorar en forma y fondo la redacción de esta tesis.

A los funcionarios del DCC: Yordy Arévalo, Teresa Huenunguir, Christián Bridevaux y Angélica Aguirre por apoyarme en la gestión de diversos temas durante mi paso por el programa.

Tabla de contenido

Índice de tablas

Índice de ilustraciones

1. Introducción

1.1. Contexto

El presente trabajo tiene su génesis en responder a una necesidad constante de la empresa Citymovil S.A. de expandir las funcionalidades de su plataforma informática de gestión y monitoreo de flota. La compañía está enfocada en el desarrollo e implementación de soluciones inteligentes de transporte [1], con más de 17 años de experiencia, cuya presencia se expande a industrias tales como: minería, transporte público y privado, vehículos de emergencia, logística y retail.

El modelo de negocio de esta empresa es ofrecer el servicio de monitoreo de vehículos en tiempo real, en base a la personalización de las aplicaciones *Controloptimo* y *Sinóptico,* con el objetivo de responder a los requerimientos de negocio particulares de cada cliente.

Las aplicaciones antes mencionadas fueron construidas de manera íntegra por la compañía. Para ello cuenta con un área de desarrollo y mantenimiento compuesta por un equipo de ingenieros y técnicos formados en disciplinas tales como informática, electrónica y electricidad. La gestión de la plataforma está a cargo del área de operaciones donde cabe destacar la labor del equipo de técnicos que realizan tareas en terreno y el equipo de soporte 24 x 7 quienes canalizan incidencias de los sistemas y gestionan las visitas a cliente.

Una solución informática para el monitoreo de vehículos, contempla un conjunto de características típicas [2], las que se enumeran a continuación:

- Visualizar la posición actual de un vehículo.
- Consultar por el historial de desplazamientos.
- Detectar e informar el paso de vehículos por lugares determinados.
- Generar alertas para situaciones anormales de conducción del vehículo.
- Monitorización de variables propias del vehículo.
- Reportes de velocidad, combustible, eventos de conducción, etc.
- Asignación de conductores para determinados trayectos.
- Despliegue de indicadores de gestión para el negocio de cada cliente.

Desde el punto de vista del sistema de monitoreo de flota, éste se puede modelar conceptualmente en 3 ambientes que se comunican sobre una red de datos TCP/IP. Los ambientes de ejecución se muestran en la figura 1.1 y pueden ser definidos como:

- **Equipo a bordo:** son todos los dispositivos y aplicaciones que existen dentro del vehículo.

- **Plataforma de monitoreo:** corresponde a los sistemas de información que se ejecutan en servidores de la compañía.
- **Centro de control:** se refiere al software que se ejecuta como parte de las aplicaciones del cliente.

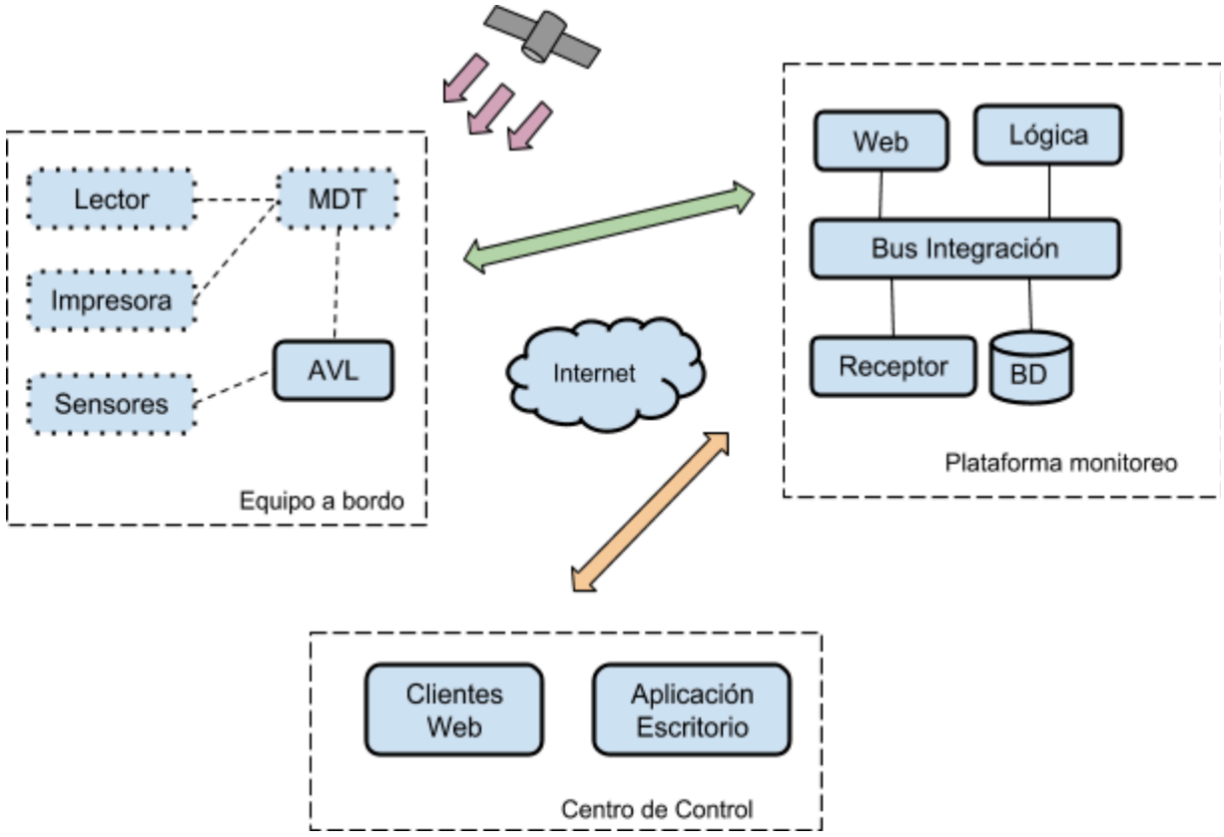

Figura 1.1: Esquema de alto nivel para solución de monitoreo de flota.
(Diagrama de autoría propia).

Uno de los componentes esenciales en la arquitectura es el dispositivo embarcado AVL (Automatic Vehicle Location). Su misión primordial es obtener los datos de georreferencia del vehículo, usando la información provista por la red satelital GPS (Global Positioning System) [3]. Para transmitir de forma periódica dicha información a los servidores, la estrategia más utilizada en la actualidad es a través de redes GPRS (General Packet Radio Service) [4] dado que la evolución de esta tecnología permite que los costos de implementación sean más bajos [5] respecto a otras soluciones, tales como mensajería SMS o redes de datos satelitales.

La información que se genera en el vehículo se recolecta de forma lógica, a través de una aplicación de recepción específica para el protocolo del fabricante del AVL y se

convierte a un formato estándar del sistema para su posterior procesamiento, análisis y almacenamiento en los servidores centrales de la plataforma de monitoreo de flota.

Para clientes que necesitan resolver problemáticas con un grado mayor de complejidad relacionadas con eventos que ocurren a bordo del vehículo, se les suele incorporar un equipo más elaborado conocido genéricamente como MDT (Mobile Data Terminal). Estos equipos disponen de pantalla y botonera para interactuar con el conductor y son capaces de ejecutar servicios embebidos para atender funcionalidades programables, además de proveer de interconectividad con otros dispositivos tales como: lectores RFID, impresoras, etc. junto con otros sensores de tipo analógico o digital. En [6] se discuten otras aplicaciones que pueden considerarse como parte de servicios orientados a la seguridad.

Como parte del software utilizado por los usuarios finales, se puede mencionar que la aplicación *Controloptimo*, opera íntegramente vía navegador web y permite hacer monitoreo de vehículos de propósito general. Por otro lado, *Sinóptico* es una aplicación de escritorio enfocada en el transporte público de buses. De acuerdo al servicio contratado por el cliente se habilitan distintos módulos de éstas, siendo el más básico la interfaz de rastreo de vehículos.

Cuando el cliente final corresponde a una organización de tamaño medio a grande, como por ejemplo una empresa de transporte, radiotaxis o un departamento de logística de una compañía, por lo general se implementa un área llamada COF (Centro de Operación de Flota) con una cantidad de operadores proporcional a la cantidad de vehículos, en donde se entrenan a usuarios específicos para cada módulo del sistema.

1.2. Problema a resolver

Con el avance de la tecnología de sistemas de navegación GPS, ha proliferado una nueva gama de dispositivos para ser usado a bordo de vehículos, cuyo aspecto se puede ver en la figura 1.2. Estos aparatos disponen de una interfaz gráfica en la que su mayor uso es el despliegue de mapas para señalar al conductor, entre otras cosas, la ubicación y el tiempo estimado de llegada a destino con información calculada de forma autónoma. Muchos fabricantes además implementan algoritmos que son capaces de ofrecer rutas optimizadas en tiempo o distancia y también la posibilidad de indicar de forma precisa puntos de interés durante el recorrido, tales como estaciones de servicio, hospitales y otros.

Un caso particular de equipos de posicionamiento con las características antes descritas, corresponde a ciertos modelos del fabricante Garmin que complementan dichos servicios con: mensajería de texto, envío de puntos de destino, recepción de alertas, administración de conductores y otros más. A través del uso de una API (Application Programming Interface) de comunicación propietaria, estos sistemas pueden ser

accedidos vía puerto USB para habilitar e interactuar con éstas últimas características [7].

Figura 1.2: Navegador GPS Garmin NUVI.
(Imagen obtenida de www.garmin.com).

La posibilidad de integrar los dispositivos antes mencionados con los sistemas de gestión de flota que ofrece la empresa Citymovil S.A., se presenta como una oportunidad de mejora altamente atractiva para el portafolio de servicios que provee la compañía. Quienes se verían beneficiados en mayor medida son clientes para los cuales es parte importante de su negocio la gestión de solicitudes de desplazamiento con los conductores, como por ejemplo en empresas de despacho y logística, radiotaxis, servicio técnico en terreno, ambulancias, arriendo de vehículos [8], etc.

Generalizando el rubro en específico al que se dedique el cliente final, en un sistema como el propuesto se pueden identificar claramente tres roles o beneficiarios:

- **Despachador / Operador:** Es el encargado de gestionar las solicitudes de desplazamiento a través de la interfaz del sistema.
- **Conductor / Equipo en terreno:** Son los usuarios que se encuentran en ruta y que reciben las instrucciones del Despachador. Ellos utilizan directamente el dispositivo de navegación GPS.
- **Administrador:** Este usuario responde generalmente a un rol supervisor. Su interés es obtener información de alto nivel sobre el cumplimiento de despachos. No siempre va a corresponder a un usuario directo del sistema

Se espera que la productividad de cada uno de estos beneficiarios pueda verse incrementada al incorporar una solución como la que se propone en esta tesis. Es por ello que se trabajará sobre la base de encuestas para medir la satisfacción de los roles y validar el impacto que tiene esta solución tecnológica en su actividad cotidiana.

Según estudios recientes [9], se estima que el mercado de aplicaciones GPS a nivel global, debería crecer un 23,7% anual para los próximos años. A nivel del mercado nacional, la oferta de soluciones basadas en tecnología de monitoreo GPS para vehículos es variada y cubre diversas industrias y problemáticas. Se estima que existen al día de hoy del orden de 35 empresas formales en este rubro que tienen presencia en nuestro país, lo cual implica un crecimiento de la oferta de cerca de 20% anual si se considera que hace 10 años en el mercado local no existían más de 5 empresas dedicadas al tema, de acuerdo a las estimaciones internas de la compañía. Por otro lado, se estima que menos del 20% de las empresas en cuestión disponen de las competencias en I+D+I para incorporar mejoras en sus productos e infraestructura de monitoreo.

Citymovil S.A. no sólo se logra diferenciar de sus pares por el nivel de conocimiento de la industria del transporte público, sino que también desde el punto de vista tecnológico, al establecer políticas de dominio absoluto de sus plataformas y la reciente asociación estratégica con empresas extranjeras para ampliar la oferta de servicios. En esa misma línea, el hecho de abordar esta oportunidad de mejora hoy en día, hace que la empresa siga explotando sus elementos diferenciadores con respecto a sus pares, ya que la madurez tecnológica de la industria nos indica que los servicios tienden a convertirse con los años en commodities, como ya ocurrió con el monitoreo GPS básico.

1.3. Solución

La solución considera el modelamiento y construcción de un sistema de software para ampliar las prestaciones de la plataforma de gestión y monitoreo de flota que ofrece la compañía Citymovil S.A.. Las funcionalidades de este módulo deben ser suficientes para que pueda conformarse un puente de información bidireccional entre usuarios *despachadores* y *conductores*, para así contribuir en la comunicación de rutas y destinos utilizando equipamiento especializado.

Desde el punto de vista técnico, los principales desafíos que se abordan para cumplir con la tarea antes descrita se resumen en:

- Diseñar la arquitectura de la solución, teniendo en consideración la integración con algunos de los módulos existentes en la plataforma de servicios GPS de la compañía.
- Desarrollar un servidor de envío y recepción de mensajes con el cual el dispositivo AVL pueda mantener un canal de comunicación confiable y así usar sus características como un puente de datos sobre redes GPRS, y por consecuencia, con el dispositivo de navegación.
- Estudiar en detalle el protocolo de integración FMI (Fleet Management Interface) [10], implementado en algunos dispositivos de navegación Garmin, para hacer uso de las funcionalidades que son de interés para este trabajo.

- Construcción de un servicio para la integración con la plataforma de gestión de flota donde se podrá hacer uso de los componentes de esta solución tecnológica.

El desarrollo en cuestión permite ofrecer un conjunto de nuevas funcionalidades a explotar por la compañía, que van desde un sistema de mensajería de texto privado hasta el envío de órdenes de servicio para desplazamientos con indicaciones de georreferenciación.

Desde el punto de vista de los roles del sistema, cada uno de ellos se puede ver beneficiado en distintos aspectos relacionados con su actividad:

- **Despachadores / Operadores:** Se espera que puedan coordinar de manera más eficiente las actividades que debe realizar el equipo en ruta, al no tener que dedicar esfuerzo innecesario en la comunicación verbal de direcciones o puntos geográficos, que muchas veces son desconocidos por ambas partes. Además se posibilita llevar un control más preciso de las órdenes de servicio asociadas a estos viajes, lo cual potencialmente ayuda a perfeccionar los procesos logísticos de cada cliente.
- **Conductores / Equipo en terreno:** El sistema al ser capaz de proveer un listado de recorridos a realizar, planificados por el despachador, deja al conductor con la principal tarea de llegar a destino. Utilizando las sugerencias del dispositivo navegador, además puede conocer el tiempo estimado de llegada. También, se puede incorporar mensajería para indicar el estado de realización de alguna tarea propia del negocio.
- **Administradores:** El beneficio para estos usuarios es el mejoramiento de indicadores de gestión internos que pudiese tener el área o empresa relacionados con el cumplimiento de los servicios de despacho. Se espera una mayor satisfacción de los clientes, lo que indirectamente también incide en la imagen de la empresa.

Para obtener una retroalimentación del funcionamiento de la solución antes incorporarlo como un nuevo servicio de la compañía, se contempla la ejecución de un plan piloto con usuarios internos, en particular en el área de soporte junto al servicio técnico en terreno. Este escenario permite que la evaluación del sistema se pueda obtener a través de indicadores derivados de los niveles de satisfacción de los usuarios, considerando la generación de estos niveles antes de la implantación del sistema y luego de un periodo de marcha blanca para comparar resultados.

El análisis económico está fuera del alcance de este trabajo, no obstante, se puede mencionar que la solución es pensada para utilizar tecnologías de software libre.

1.4. Objetivos

1.4.1. Objetivo General

- Integrar con efectividad los dispositivos de navegación *Garmin nüvi* con la plataforma *Controloptimo* de la empresa *Citymovil S.A.* para facilitar la comunicación y coordinación entre usuarios despachadores y conductores.

1.4.2. Objetivos específicos

- Mejorar el nivel de satisfacción de usuarios despachadores en un 20% al incorporar procesos y herramientas para facilitar la tarea de gestionar servicios en terreno.
- Aumentar en un 15% la satisfacción de usuarios conductores al entregar tecnología a bordo que le permita optimizar sus tiempos de desplazamiento.
- Conseguir un incremento del 10% en la satisfacción de usuarios administradores debido al aumento en el cumplimiento de los servicios de despacho.

La definición de estos objetivos se elaboró bajo un criterio terminal, es decir, enfocado en la experiencia hacia los usuarios más que en objetivos metodológicos que estuviesen centrados en abordar la construcción del sistema. Con respecto a los niveles de satisfacción esperados, se puede mencionar que su elaboración está basada en el criterio experto obtenido desde la empresa, aprovechando el conocimiento acumulado a través de los años en la construcción de software para la industria del transporte.

1.5. Metodología

Para cumplir con los objetivos descritos en el apartado anterior, este trabajo considera el desarrollo de una solución de software. Como se puede esperar de un trabajo de estas características, la primera fase de análisis tiene un importante componente de investigación de la tecnología subyacente. Las conclusiones que se obtienen en esa etapa son la base para mejorar la arquitectura preliminar de software y de sistemas. Durante la fase de diseño, junto con la definición del modelo de datos a usar, se estudia y especifica el flujo de información entre los componentes involucrados. Además, se definen pruebas que el sistema debe cumplir.

Paso seguido se da comienzo la construcción de los módulos de software, donde se identifican a priori tres componentes fundamentales:

- Servidor de comunicación con los equipos AVL.
- Bibliotecas de traducción del protocolo FMI.
- Servicios para la integración con sistema externo.

Al finalizar la etapa de construcción, se realiza una primera evaluación al grupo de usuarios, con el objeto de tener una base comparativa de su nivel de satisfacción con respecto al trabajo final. Estos usuarios de prueba, como se ha mencionado en apartados anteriores, corresponden al personal del área de soporte de la compañía y a técnicos que prestan servicios en terreno.

En una fase posterior se realiza la implementación del sistema en un ambiente piloto, donde además se espera instalar el equipamiento de hardware en vehículos de la compañía para validar el funcionamiento en una situación de operación real. Al cerrar esta etapa, se realiza una segunda evaluación a los usuarios para generar los indicadores de satisfacción final.

1.6. Plan

Las etapas planteadas con anterioridad consideran una serie de actividades (Ver tabla 1.1) que se pueden tipificar según la fase en que se implementan, la complejidad y el tiempo estimado que tomaría el llevarlas a cabo.

Según la estimación inicial, basada en criterio experto, el esfuerzo para llevar a cabo este proyecto se calcula en 180 días de trabajo efectivo.

El criterio para definir si una actividad corresponde a una complejidad alta, media o baja, tiene relación con el tiempo asociado a su ejecución:

- Baja: de 1 a 3 días.
- Media: de 4 a 12 días.
- Alta: de 13 a 20 días.

Etapa	Actividad	Complejidad	Duración estimada (días)
A	Estudio de integración de dispositivo AVL	media	10
A	Estudio protocolo FMI de Garmin	alta	15
A	Análisis y modelamiento de componentes del negocio	media	8
A	Análisis de interoperabilidad con otros sistemas	baja	3
D	Definición de la arquitectura y tecnologías	media	6
D	Diseño de módulos y APIs de comunicación	media	12
D	Modelamiento Base de datos	media	8
D	Modelamiento de integración con sistemas externos	media	12
I	Desarrollo de API para gestión de datos Garmin	alta	15
I	Desarrollo módulo servidor FMI	media	12
I	Desarrollo bibliotecas para protocolo FMI	alta	20
I	Desarrollo de integración con otros sistemas	alta	20
I	Implementación en ambiente de pruebas	media	5
E	Definición de encuesta de satisfacción	media	12
E	Toma de encuesta inicial	baja	1
P	Pruebas de comunicación de servidor con AVL	baja	2
P	Pruebas de módulos web	baja	3
P	Pruebas de integración entre sistemas	baja	3
P	Piloto de validación con cliente de pruebas	media	5
E	Toma de encuesta final	baja	1
E	Consolidación de resultados de evaluación	baja	2
E	Análisis encuesta	media	5

Tabla 1.1: Listado de actividades, su complejidad y duración estimada.
(Tabla de autoría propia).

Las fases en el proceso de implementación de la tabla 1.1, se definen de la siguiente manera:

- **Etapa A (Análisis):** actividades relacionadas con el estudio de las tecnologías subyacentes y el estudio de las entidades de un sistema de despacho.
- **Etapa D (Diseño):** actividades que corresponden al modelamiento a nivel de componentes y el diseño de la interacción entre ellos para dar forma a la solución buscada.
- **Etapa I (Implementación):** actividades donde se realiza la construcción de los distintos módulos y subsistemas.
- **Etapa P (Pruebas):** actividades que permiten poner a prueba el sistema con el objetivo de encontrar defectos.
- **Etapa E (Evaluación):** tareas donde se desarrolla la evaluación del sistema en base a encuestas a los interesados.

El proceso de construcción se ejecuta de forma iterativa e incremental, para tener resultados o hitos con mayor periodicidad.

2. Marco teórico

2.1. Conceptos para integración con equipamiento a bordo

La arquitectura del sistema, está compuesta por elementos de hardware y software. Los dispositivos de hardware en este contexto, son aquellos que se instalan en el vehículo, tales como:

- AVL Skypatrol
- Garmin nüvi

Los componentes de software, por otro lado, tienen que ver con servicios de Backend que permiten enviar, recibir, interpretar y desplegar la mensajería desde y hacia el equipo Garmin. A este conjunto de servicios se les denomina “Módulo Garmin”.

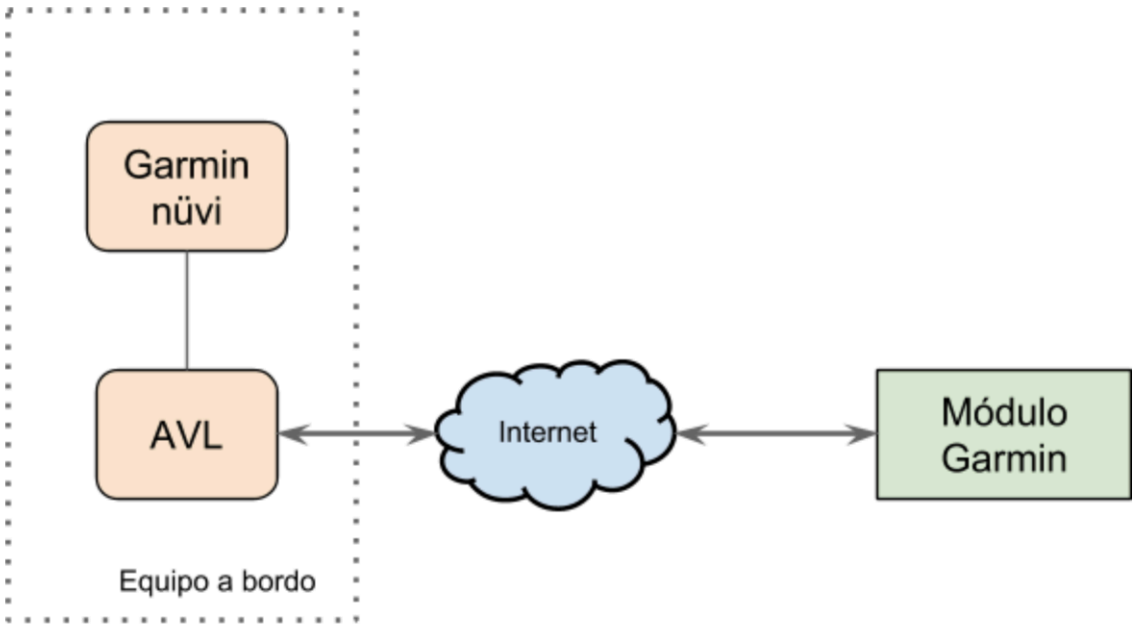

Figura 2.1: Esquema de integración para equipos a bordo.
(Diagrama de autoría propia).

De acuerdo al esquema lógico de la integración con el equipamiento embarcado, señalado en la figura 2.1, se puede desprender que el Módulo Garmin debe ser capaz de proveer algún mecanismo que le permita al equipo AVL utilizar la red Internet para comunicarse con éste. El servidor de recepción antes señalado es uno de los subsistemas que son parte de la solución propuesta.

Por otro lado, el equipo AVL al disponer de un modo especial de funcionamiento denominado PAD (Packet Assembler / Disassembler), permite que su puerto serie físico RS232 replique toda la información en el servidor TCP/IP remoto. El equipo Garmin hace uso del puerto serie donde se crea un enlace de forma lógica y bidireccional con la parte del servidor capaz de traducir el protocolo FMI (propietario de Garmin).

Cabe destacar que el equipo AVL paralelamente a la funcionalidad señalada en esta sección, puede ser configurado para generar sus propios eventos o mensajes relacionados a la georreferencia y elementos de alerta, pero estos se capturan con otro tipo de receptor de datos.

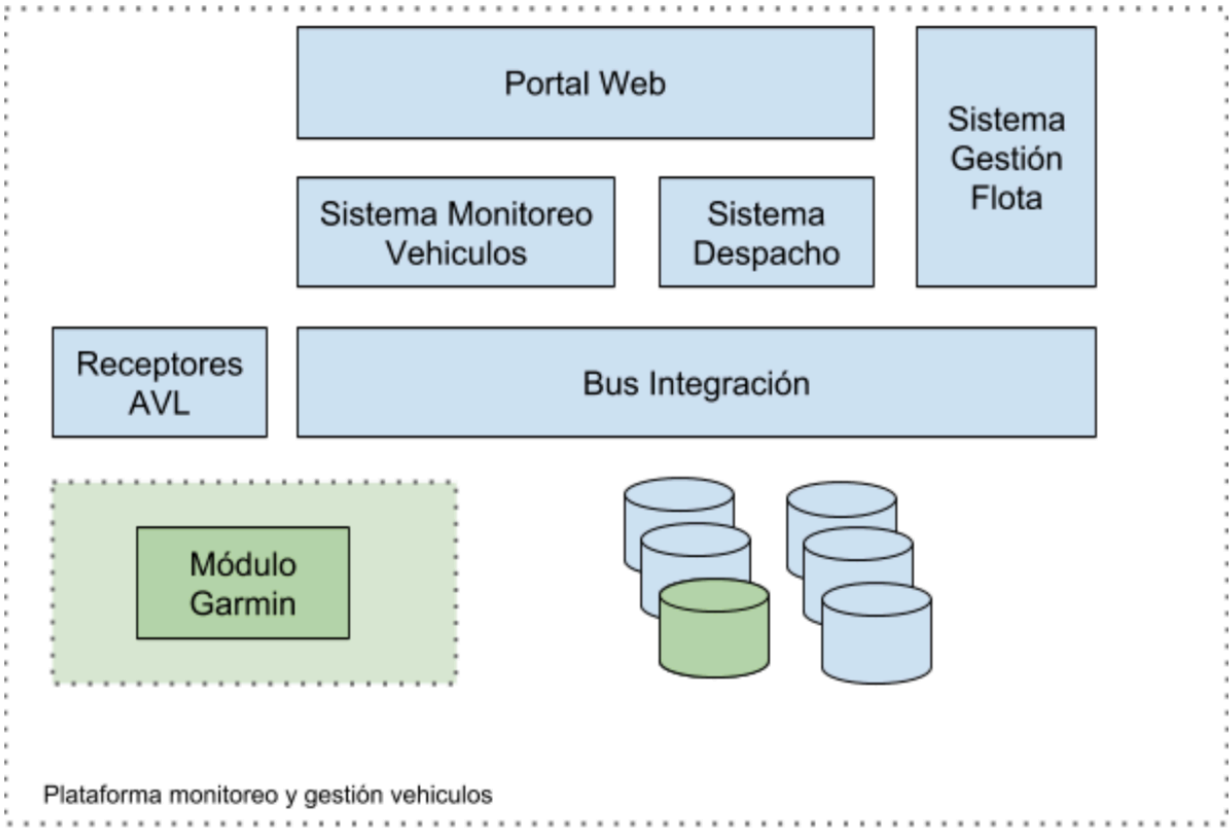

Figura 2.2: Aplicaciones o sistemas que componen la plataforma monitoreo de vehículos. (Diagrama de autoría propia).

El esquema general de los componentes más importantes de la plataforma de monitoreo se presenta en la figura 2.2. La descripción de los servicios mencionados es la siguiente:

- **Portal Web:** Aplicación orientada al usuario final que permite la visualización y configuración de vehículos, conductores y otros objetos gestionados por el sistema, así como también la presentación de indicadores del negocio para cada cliente.

- **Sistema Monitoreo Vehículos:** Conjunto de servicios de Backend enfocados en proveer de información al Portal Web relativa al seguimiento de los vehículos y al procesamiento de alertas.
- **Aplicación de Gestión de Flota:** Sistema que permite visualizar el cumplimiento de rutas e itinerarios para el transporte público.
- **Aplicación de Despacho:** Sistema que permite generar órdenes de servicio para clientes de transporte público.
- **Bus de Integración:** Aplicación de mensajería orientada en proveer servicios de integración basados en colas y tópicos.
- **Receptores AVL:** Son un conjunto de servicios que reciben la mensajería de posicionamiento de los AVL.
- **Base de Datos:** Sistemas orientados al almacenamiento de la información de las aplicaciones antes mencionadas.

El Módulo Garmin se incorpora como uno más dentro de los servicios disponibles de la plataforma, pudiendo interactuar de manera automática, por ejemplo, con el sistema de despachos para obtener los viajes programados para un determinado cliente. En el capítulo 4 se aborda con mayor detalle el diseño detallado y la comunicación entre los componentes involucrados.

2.2. Estudio de dispositivo AVL

El equipo Skypatrol TT8750 es un dispositivo de Localización Automática Vehicular (AVL) que combina en un módulo compacto un módem GSM/GPRS y un módulo receptor de Posicionamiento Global por Satélite (GPS). El equipo se completa funcionalmente con sus antenas externas respectivas.

La figura 2.3 muestra una imagen del aspecto físico del dispositivo. Este equipo en particular es uno de los más simples y versátiles que se pueden encontrar en el mercado, lo cual ha hecho que sea ampliamente adoptado por empresas del rubro. Las características más destacadas que posee este equipo [11] se enuncian a continuación:

- Rastreo GPS
- Entradas/Salidas multipropósito analógicas y digitales (Telemetría)
- Configuración inalámbrica (OTA)
- Almacenamiento de Geocercas
- Generación de alertas
- Odómetro virtual
- Operación cuatri-banda GSM (850/900/1800/1900 MHz)

Figura 2.3: Equipo AVL SkyPatrol.
(Imagen obtenida de www.navixy.com).

El método de configuración para programar este equipo es a través del envío de comandos “AT” directamente a su puerto serie a través de una sesión de terminal. Este dispositivo también soporta programación inalámbrica “Over The Air” (OTA), pero no se utilizará en este trabajo.

Comando	Descripción
ATI	Muestra información del fabricante (Enfora, Inc.)
AT&F	Restaura las configuraciones de fábrica.
AT&W	Almacena la configuración a memoria.
AT$RESET	Para forzar un reinicio del equipo.
AT+CGDCONT=1,”IP”,”apn”	Establece el APN a utilizar.
AT%CGPCO=1,”username, password”,0	Establece el usuario y password
AT$AREG=2	Para habilitar el registro automático a la red GPRS.
AT+CREG?	Verificar el status de la red GSM.
AT%CGREG?	Verifica el status de la red GPRS.
AT$NETIP?	Obtiene la dirección IP asignada por la red GPRS.

Tabla 2.1: Ejemplo de comandos para configuración de AVL.
(Tabla generada a partir de Guía de Usuario SkyPatrol TT8750 [11]).

A modo de ejemplo, en la tabla 2.1 se revisan algunos comandos básicos utilizados durante una configuración inicial del equipo AVL.

Las configuraciones mínimas que son necesarias para habilitar este dispositivo son:

- Creación de un identificador de equipo
- Establecimiento de la red de datos GPRS
- Configuración de IP y puerto para servidor de monitoreo GPS
- Habilitación de envío de mensajes de posicionamiento GPS (opcional)
- Configuración de IP y puerto para servidor de modo PAD

En el anexo A, se señalan los comandos utilizados para generar la configuración del equipo en modo PAD [12].

2.3. Estudio del Protocolo FMI de Garmin

El equipo de navegación Garmin que se utiliza, corresponde al modelo nüvi 2495. Este dispositivo, tiene la característica de interactuar con aplicaciones externas usando una API propietaria a través del puerto USB. La API de esta gama de dispositivos, también conocida como interfaz FMI, implementa protocolos [13] para la ejecución de comandos sobre el equipo de navegación, de tal forma que se pueda acceder remotamente a la información de coordenadas de destino, notificaciones de texto y datos derivados del cálculo de rutas, como el tiempo estimado de arribo (ETA), entre otras funciones que dependen de la versión del protocolo. En el caso del nüvi 2495 la versión soportada de FMI es la 2.7 tal como se señala en [14]. Esta versión representa un conjunto de características resumidas en la tabla 2.2.

Según lo que señala Garmin en su sitio orientado a desarrolladores, el foco de esta compañía no está en el desarrollo de aplicaciones de monitoreo y gestión de flota, por lo que le entrega a la comunidad las herramientas para interactuar con las funcionalidades de sus equipos. Estas herramientas se conocen como “FMI Developer Kit”, las cuales consisten en códigos de ejemplo y documentación que detalla el protocolo FMI y su integración [15].

En cuanto a las características que definen al protocolo FMI de Garmin, se puede desprender de la documentación [16] y [17] lo siguiente:

- Modelo de arquitectura Cliente – Servidor
- Protocolo de comunicación en 2 capas: física/enlace y aplicación
- Envío de datos serial, con ACK y CRC
- Trama de datos de largo variable

Protocolo	Descripción
A602	(no especificado)
A603	• Gestión de detenciones. • Tiempo estimado de llegada (ETA)
A604 (en desuso)	• Mensajes de texto libre • Mensajes y respuestas predefinidas • Status del conductor
A605	• Status del enlace de comunicaciones • Gestión de protocolos utilizados
A606	• FMI en modo seguro
A607	• Mensaje de texto abierto • Mensajes predefinidos • Puntos de ruta • Status del conductor
A608	• Alertas de velocidad
A609	• Reinicio remoto

Tabla 2.2: Resumen de funcionalidades soportadas por protocolo.
(Tabla basada en referencias [13] y [14]).

Para implementar una biblioteca que permita la comunicación con el dispositivo Garmin, se requiere analizar cómo funciona la estructura binaria de datos FMI a bajo nivel.

En la tabla 2.3 se presenta el formato de la trama de datos serial de consulta al equipo y en la tabla 2.4 se presenta la estructura de una respuesta (ACK/NAK). Tal como se puede apreciar de ambas tablas, los protocolos orientados al byte, como el FMI, incorporan una serie de caracteres de relleno para marcar el inicio (DLE) y fin (DLE, ETX) de un mensaje, además de incorporar algún medio de comprobación de la integridad de la información a través de uno más bytes para el “Checksum”.

Número de Byte	Descripción	Notas
0	Caracter Escape	Caracter ASCII DLE (decimal 16)
1	Packet ID	Identificador del tipo de paquete
2	Tamaño del mensaje de aplicación	Número de bytes del paquete de datos
3 al n-4	Datos del mensaje de aplicación	Largo variable de 0 a 255 bytes
n-3	Checksum	Complemento a 2 de la suma de todos los bytes desde el byte 1 al n-4 (Final del payload)
n-2	Caracter Escape	Caracter ASCII DLE (decimal 16)
n-1	Fin de Texto	Caracter ASCII ETX (decimal 3)

Tabla 2.3: Formato de la trama de datos serial FMI.
(Tabla obtenida de referencia [17]).

Número de Byte	Descripción	Notas
0	Carácter Escape	Carácter ASCII DLE (decimal 16)
1	Packet ID	Carácter ASCII ACK/NAK (decimal 6 o 21 respectivamente)
2	Tamaño del mensaje de aplicación	Valor 2
3	Datos del mensaje de aplicación	Identificador Packet ID del mensaje reconocido
4	NULL	Valor 0
5	Checksum	Complemento a 2 de la suma de todos los bytes desde el byte 1 al 4
6	Carácter Escape	Carácter ASCII DLE (decimal 16)
7	Fin de Texto	Carácter ASCII ETX (decimal 3)

Tabla 2.4: *Formato de la trama de respuesta FMI.*
(Tabla obtenida de referencia [17]).

Para ejemplificar lo descrito anteriormente, se analiza el primer mensaje que debe ser enviado al equipo Garmin para habilitar el uso del protocolo FMI. De acuerdo con la especificación del fabricante [11], este comando se denomina “Enable Fleet Management Protocol”. Si no se ejecuta esta instrucción al inicio de la sesión, ningún otro comando será interpretado por el equipo. En la figura 2.4 se puede ver la instrucción en formato texto.

PacketID: **0xA1**
PayloadSize: **0x0E** (14)

Payload
FMIPacketID: 0x0000
FeatureCount: 5
Feature: Unicode is enabled (1)
Feature: A607 Support is enabled (2)
Feature: Driver Password Support is disabled (10)
Feature: Multiple Drivers is disabled (11)
Feature: AOBRD Support is disabled (12)

Figura 2.4: Instrucción para habilitar interfaz FMI representada como formato texto. (Imagen de autoría propia).

Al codificar la instrucción antes mencionada al formato binario, resulta la trama de mensaje presentada en la figura 2.5. Se han coloreado los campos “Packet ID” y “Payload Size” para identificarlos dentro del mensaje FMI. La línea superior se agrega solo para un mejor entendimiento de la ubicación relativa de los bytes en la trama del mensaje.

[00]	[01]	[02]	[03]	[04]	[05]	[06]	[07]	[08]	[09]	[10]	[11]	[12]	[13]	[14]	[15]	[16]	[17]	[18]	[19]
10	A1	0E	00	00	05	00	01	80	02	80	0A	00	0B	00	0C	00	28	10	03

Figura 2.5: Instrucción para habilitar interfaz FMI en formato binario. (Imagen de autoría propia).

La trama de bytes en respuesta al comando anterior generada por el dispositivo, se muestra en la figura 2.6. La respectiva traducción para el mensaje de respuesta, de acuerdo a lo especificado en la tabla 2.4, se presenta en la figura 2.7.

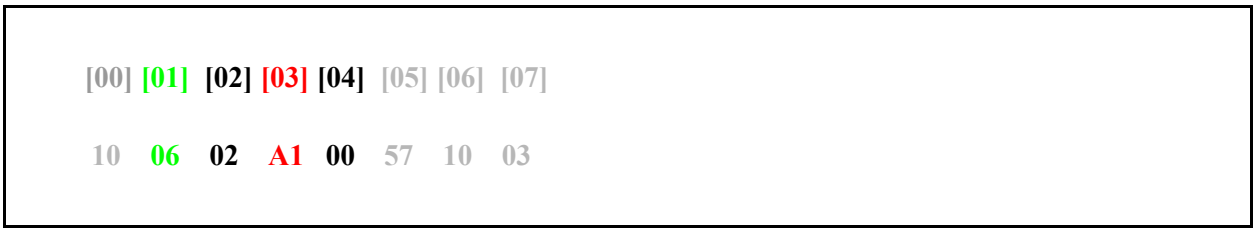

Figura 2.6: Respuesta del equipo a instrucción para habilitar interfaz FMI.
(Imagen de autoría propia).

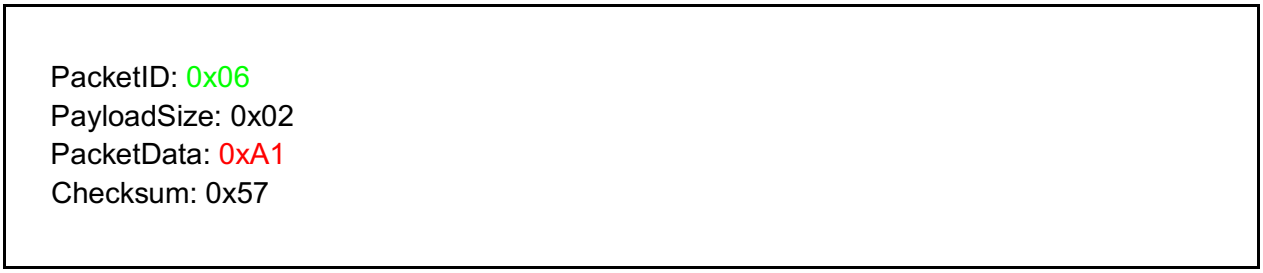

Figura 2.7: Respuesta a instrucción para habilitar interfaz FMI en formato binario.
(Imagen de autoría propia).

En el anexo B, se presentan algunos de los comandos FMI y su estructura que fueron parte de la realización de este trabajo.

3. Requisitos del sistema

3.1. Componentes del negocio

El objetivo de esta sección es obtener un modelo de dominio del problema con las entidades de negocio más comunes y representativas que interactúan en una oficina de gestión de despachos. Esta unidad organizacional se puede encontrar comúnmente en empresas de reparto, radiotaxis, servicio técnico en terreno, ambulancias o en una empresa de servicio de arriendo de vehículos, por nombrar sólo algunos ejemplos.

Los motivos para generar una solicitud de despacho son variados y responden a la necesidad de cada uno de los casos antes mencionados. Para realizar un análisis basado en el problema general, se toman en cuenta el proceso que realizan tres clientes típicos:

- Servicio técnico en terreno.
- Sistema de reparto de empresa de retail.
- Sistema de solicitud de radiotaxis.

3.1.1. Caso de estudio 1: Servicio técnico en terreno

El primer caso a revisar representa una solicitud de visita a terreno para un servicio técnico que realiza visitas a cliente. Las tareas que comúnmente se realizan bajo este contexto son las siguientes:

1. El cliente declara a la mesa de soporte un desperfecto en un equipo relacionado con el servicio ofrecido por la empresa.
2. Se evalúa si la incidencia es efectiva y se coordina con el cliente una fecha tentativa para la visita.
3. Ya sea de forma manual o con alguna herramienta de optimización, se debe generar una planificación para asignar las tareas al personal disponible. El resultado de esta tarea debe ser una Orden de Servicio que incorpore la información de: cliente, dirección de visita, motivo visita, horario visita, cantidad de personal requerido y tiempo estimado de la actividad.
4. Se le comunica al personal asignado la tarea planificada, indicando la información de la orden de servicio.
5. El personal asignado debe indicar el cumplimiento de la orden de servicio.

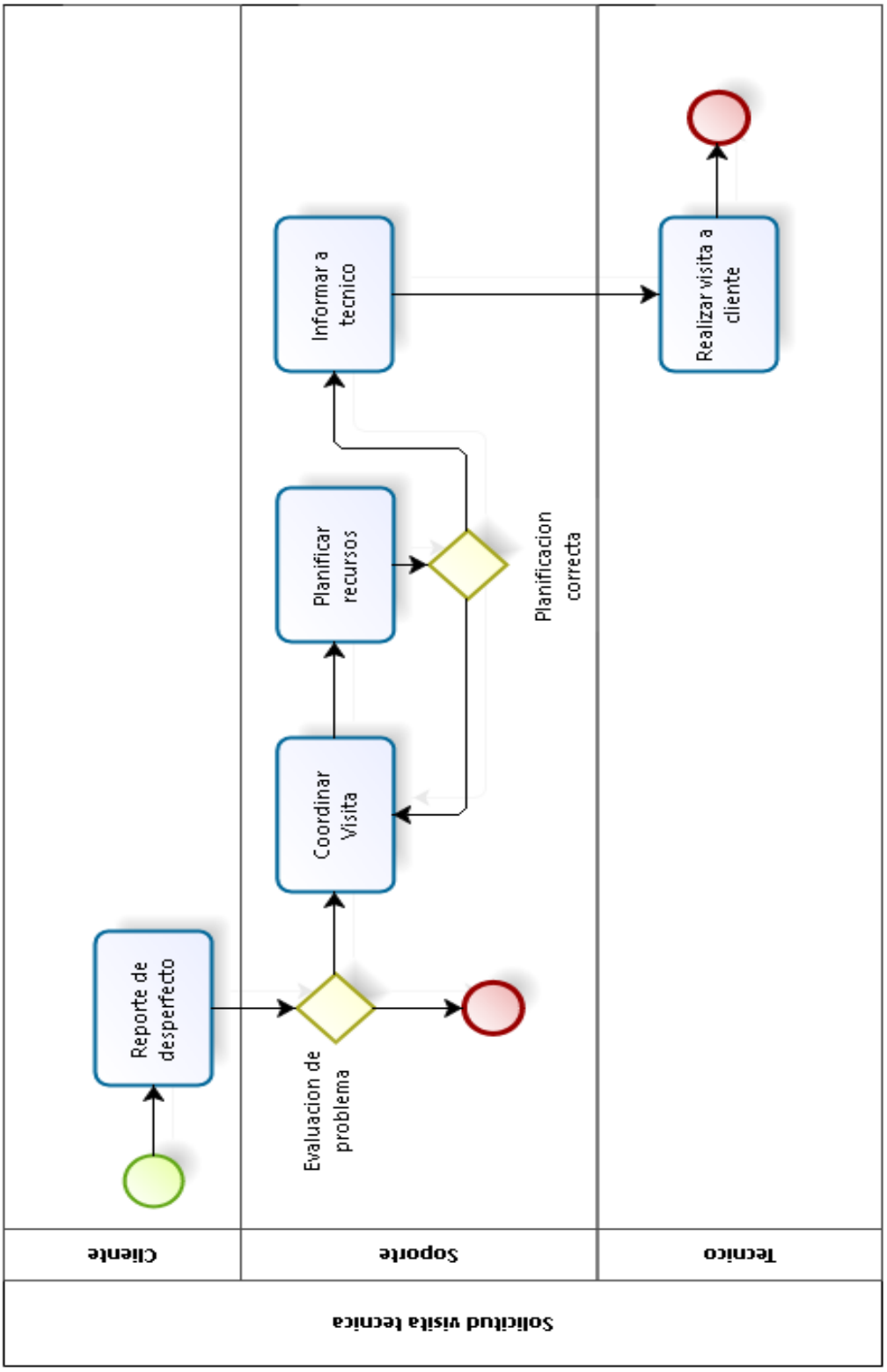

Figura 3.1: Proceso simplificado para solicitud de visita técnica.
(Imagen de autoría propia).

La figura 3.1 muestra a través de un diagrama de proceso, una simplificación de las actividades identificadas para una solicitud de visita técnica. Con el levantamiento de este proceso, las entidades del negocio que surgen para el análisis son:

- Cliente
- Lugar de la visita
- Planificación
- Orden de servicio
- Técnico en terreno (conductor del vehículo)
- Vehículo

Más adelante en este capítulo estas entidades serán consideradas para buscar un patrón común entre los distintos casos de estudio.

3.1.2. Caso de estudio 2: Sistema de reparto de empresa retail

El segundo tipo de despacho analizado, corresponde al que ejecuta una empresa de reparto para el retail. En este caso, se evalúa el proceso cuando el cliente ha comprado un producto en tienda y se pide la entrega en su domicilio. Las tareas a realizar son las siguientes:

1. El cliente visita la tienda de retail y compra un artículo con despacho a domicilio.
2. El personal de ventas evalúa el stock en el sistema de la empresa de retail e ingresa una reserva del producto.
3. El personal de ventas solicita la creación de una guía de despacho, donde se asocia la compra a la reserva.
4. Área de logística genera una programación y la ruta a seguir de acuerdo al sector geográfico.
5. Se carga el camión con los productos y comienza la ruta de entrega.
6. Una vez en el destino, el transportista obtiene la guía de despacho aprobada por el cliente, para dar cumplimiento con la entrega del producto.

El proceso simplificado para generar un despacho realizado por una tienda de retail se presenta en la figura 3.2. Análogamente al caso de estudio 1, se pueden identificar un conjunto de entidades de negocio que se utilizarán para el modelamiento del problema:

- Cliente
- Punto de entrega
- Guía de despacho
- Itinerario o programa
- Transportista
- Camión
- Despachador o administrador área logística

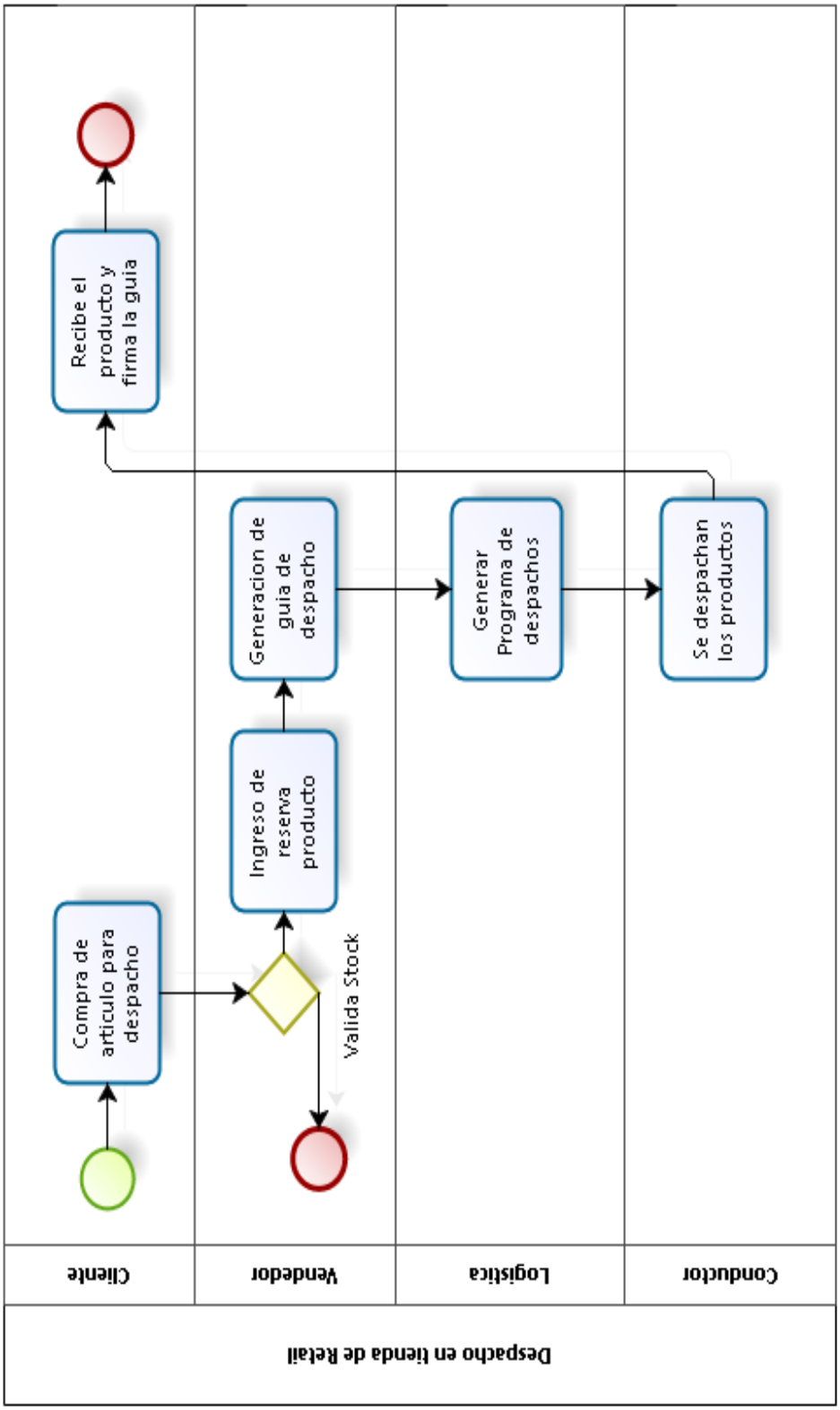

Figura 3.2: Proceso simplificado para despacho en tienda de retail.
(Imagen de autoría propia).

3.1.3. Caso de estudio 3: Sistema de solicitud de radiotaxis

Un tercer caso de estudio es el sistema de solicitud de móviles en una empresa de radiotaxis. Las tareas que se ejecutan, que permiten modelar el problema son:

1. El cliente llama a la central solicitando un móvil
2. El telefonista le solicita sus datos, la dirección de recogida y de destino e ingresa la orden al sistema de coordinación.
3. El coordinador identifica a un radiotaxi que pueda cumplir con el servicio a la hora señalada.
4. El taxista recibe la indicación para ir en búsqueda del pasajero.
5. Cuando el vehículo llega al punto de origen del viaje, la central llama al pasajero para indicar el taxi lo espera.

Las entidades de negocio que se identifican en este caso de estudio son:

- Cliente
- Punto de recogida y destino
- Orden de servicio para radiotaxi
- Recorrido del viaje
- Conductor
- Taxi
- Coordinador

En la siguiente sección se intenta modelar conceptualmente el problema general de trabajar con información relacionada a sistemas de despacho, utilizando las entidades que han surgido en estos casos de estudio.

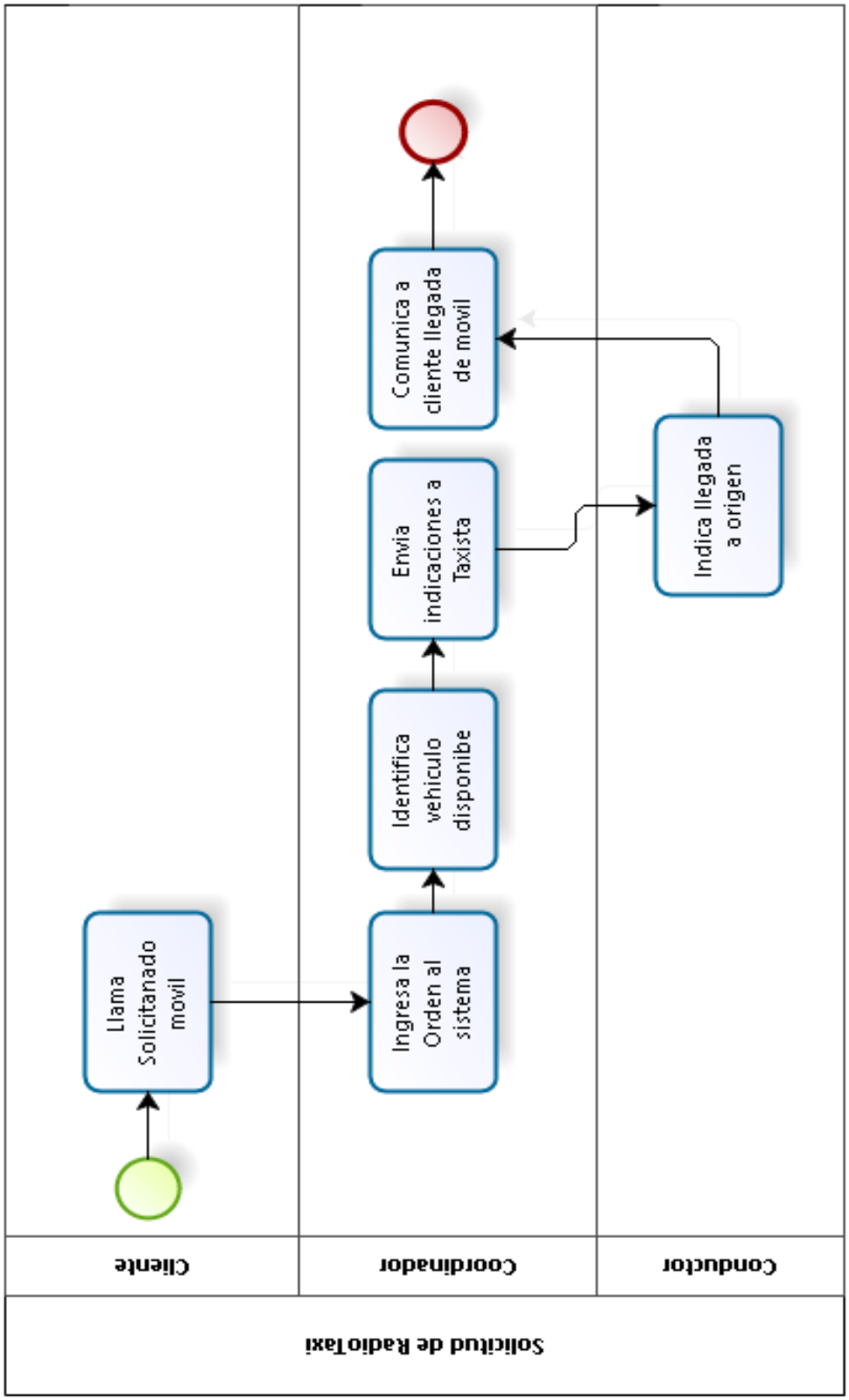

Figura 3.3: Proceso simplificado para solicitud de radiotaxi.
(Imagen de autoría propia).

3.2. Análisis y modelamiento

Cada uno de los casos de estudio vistos anteriormente, permite obtener un conjunto nuevo de entidades a considerar en el desarrollo de una aplicación orientada a apoyar la labor de despacho.

Si se ordenan las entidades de negocio acorde con la equivalencia de los términos para distintos tipos de cliente, se obtiene el siguiente listado:

- clientes / usuarios
- conductor / técnico en terreno / transportista
- vehículo / camión
- orden de servicio / guía de despacho / factura
- ruta / plan de recorrido / programación / itinerarios
- despacho / asignación de trabajo
- dueño / gerente / administrador
- coordinador / despachador

Desde el punto de vista del dominio del problema y aplicando terminología genérica para referirse a las entidades de negocio, se puede representar el modelo conceptual del sistema de despacho tal como se señala en la figura 3.4. En este listado de entidades también identifican 2 de los beneficiarios del sistema expuestos en capítulos anteriores.

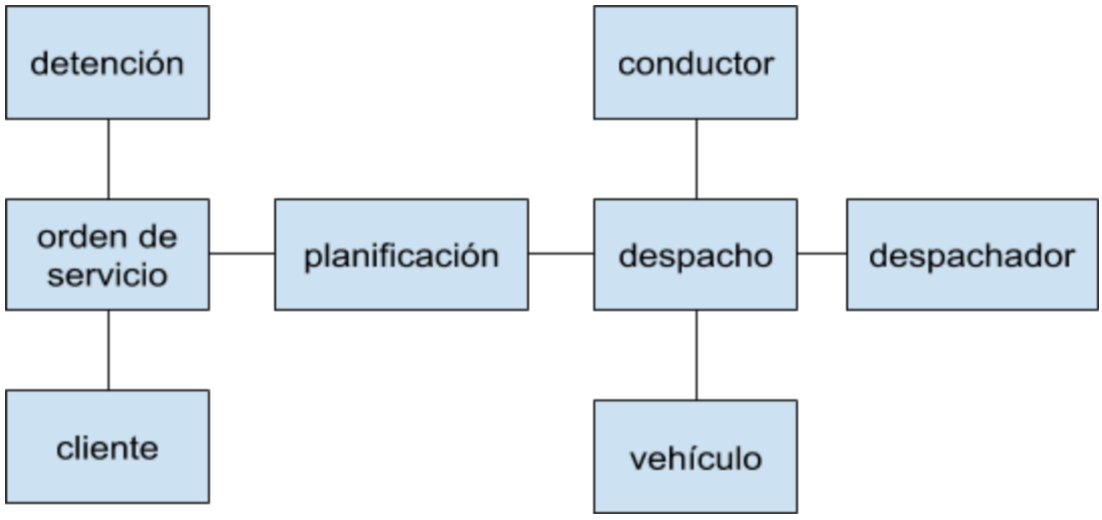

Figura 3.4: Diagrama de dominio de los objetos del sistema de despacho.
(Diagrama de autoría propia).

En la medida que se requiera una mayor especialización del proceso de despacho del cliente final, se debe representar dicha complejidad en el modelo antes presentado.

El sistema de despacho, en el contexto de la solución, es un componente auxiliar al sistema, encargado de gestionar los viajes programados para la flota de vehículos. Este debe cumplir, en términos funcionales, con los siguientes procesos a realizar:

- Administración de solicitudes de viaje
- Gestión de órdenes de trabajo
- Asignación de tareas a conductores

En los próximos capítulos se revisan los elementos más importantes que conforman la integración con los objetos de negocio a más bajo nivel.

3.3. Interoperabilidad con otros sistemas de la compañía

El Módulo Garmin a diseñar debe ser capaz de interactuar con otros sistemas de la compañía, tales como:

- Sistema de gestión de flota
- Sistema de monitoreo de vehículos
- Aplicación de despacho
- Aplicación de cumplimiento de programación

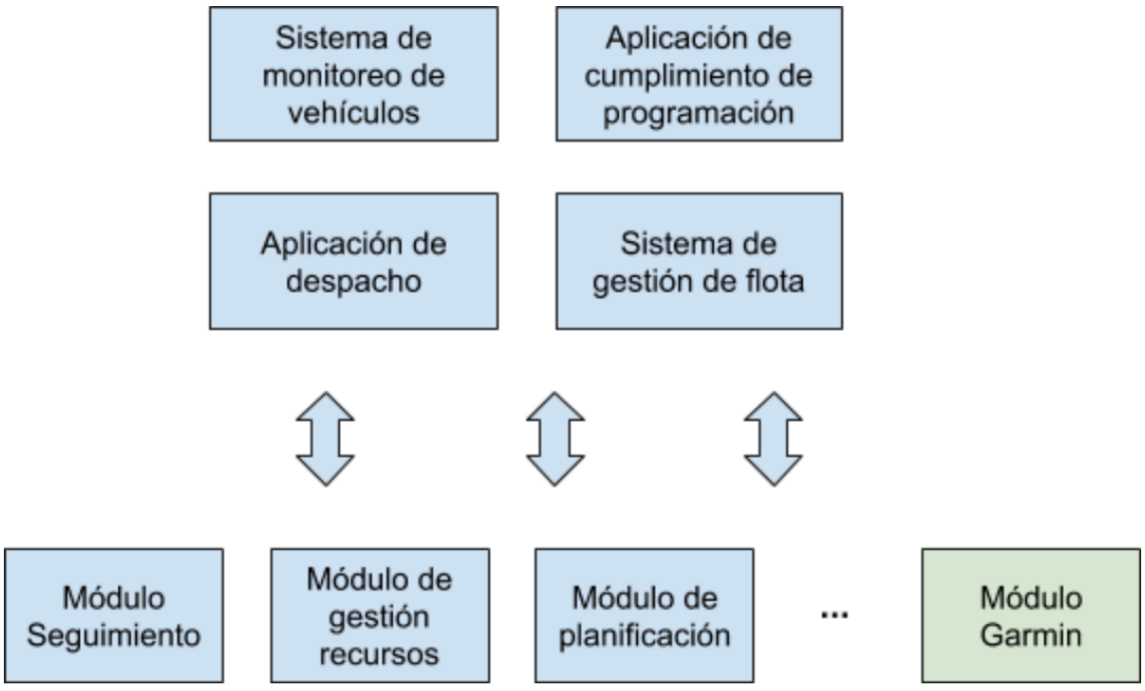

Figura 3.5: Sistemas que pueden hacer uso de la solución.
(Diagrama de autoría propia).

El requisito anterior se presenta como una necesidad muy importante para la empresa, ya que se espera que este trabajo permita potenciar los sistemas existentes y así darle mayor valor a la oferta de servicios entregados a los clientes.

Actualmente, como proceso de modernización de la arquitectura de integración de la compañía, se está impulsando la migración de la plataforma de aplicaciones hacia un esquema de interoperabilidad basado en micro servicios. Bajo este argumento, se plantea el desarrollo del sistema de comunicación con el Módulo Garmin siguiendo la misma línea.

3.4. Otros requisitos no funcionales

Algunos aspectos no funcionales extra a considerar para el modelamiento de la aplicación:

- El sistema debe ser capaz de manejar de forma eficiente la comunicación simultánea con múltiples dispositivos Garmin. (Eficiencia)
- El sistema debe tener un buen desempeño bajo las condiciones de trabajo de los usuarios. (Desempeño)
- El sistema debe ser tolerante a las intermitencias en los canales de datos inalámbricos GPRS. (Confiable)

En el anexo C se presenta el resumen de los requisitos funcionales y no funcionales que se desprenden de este trabajo.

3.5. Encuesta de satisfacción y prueba piloto

El desarrollo de este trabajo considera evaluar el sistema a través de elementos subjetivos relacionados con la percepción de los usuarios durante el transcurso de la prueba piloto. Para ello se diseña una encuesta enfocada en medir la satisfacción de los usuarios directos e indirectos en varios aspectos relacionados con su quehacer diario, primeramente sin hacer uso del sistema de apoyo a las tareas despacho y luego utilizando la aplicación.

Se elige realizar una evaluación de tipo encuesta porque dentro de las herramientas más conocidas para este tipo de tareas (por ejemplo: focus group, investigación contextual, etc), se estima que es la que más se ajusta al contexto de la organización y sobre todo por temas de disponibilidad de los usuarios.

La estructura de la encuesta considera preguntas generales y específicas del proceso de despacho para todos los roles. La sección general de la encuesta contiene 16 preguntas cuyo puntaje va de 1 a 5 para cada una. Por lo tanto el puntaje de esa sección pudiese fluctuar de 16 a 80. En las secciones específicas para cada rol, se aplica la misma escala,

no obstante el análisis de los datos se debe realizar de forma separada. Las puntuación de la sección para el Administrador va de 4 a 20 y, en el caso del Despachador y Conductor, de 2 a 10. El detalle de las preguntas de la encuesta se encuentra en el anexo D.

El cálculo para medir el incremento en el nivel de satisfacción se realiza simplemente evaluando la suma de puntos obtenidos en encuesta inicial para las preguntas generales y específicas, y comparando ese puntaje con la encuesta final para cada usuario.

Desde el punto de vista metodológico, el tipo de evaluación a realizar es de tipo sumativa ya que permite comparar el antes y el después de la implementación del sistema para validar el impacto de la solución tecnológica en la actividad cotidiana de los usuarios. La utilización de una encuesta además apoya directamente este propósito.

El ambiente piloto donde se realizan las pruebas y donde se evalúa la satisfacción de los usuarios, corresponde a la oficina de soporte de la misma empresa Citymovil S.A., donde a diario se debe coordinar la visita a terreno con técnicos, ingenieros y fuerza de venta, además de gestionar los vehículos de la empresa a sus solicitantes.

4. Diseño de la solución

En esta sección se discuten los aspectos técnicos relacionados con el diseño del sistema. Se revisan en primera instancia los elementos que permiten modelar el “Backend” de la solución, tanto del punto de vista de la integración con el dispositivo embarcado y la mensajería con éste, para luego abordar el diseño del “Frontend”.

4.1. Definición de arquitectura del Módulo Garmin

Con el objetivo de simplificar la cantidad de componentes que interactúan en esta arquitectura, por el lado de la plataforma de gestión de flota, se considera para el análisis sólo dos herramientas: el sistema de monitoreo de vehículos y el sistema de despacho. Definiendo la interacción con ambas aplicaciones, se puede diseñar las funcionalidades de mensajería y envío de itinerario de viajes al equipo Garmin de manera más sencilla.

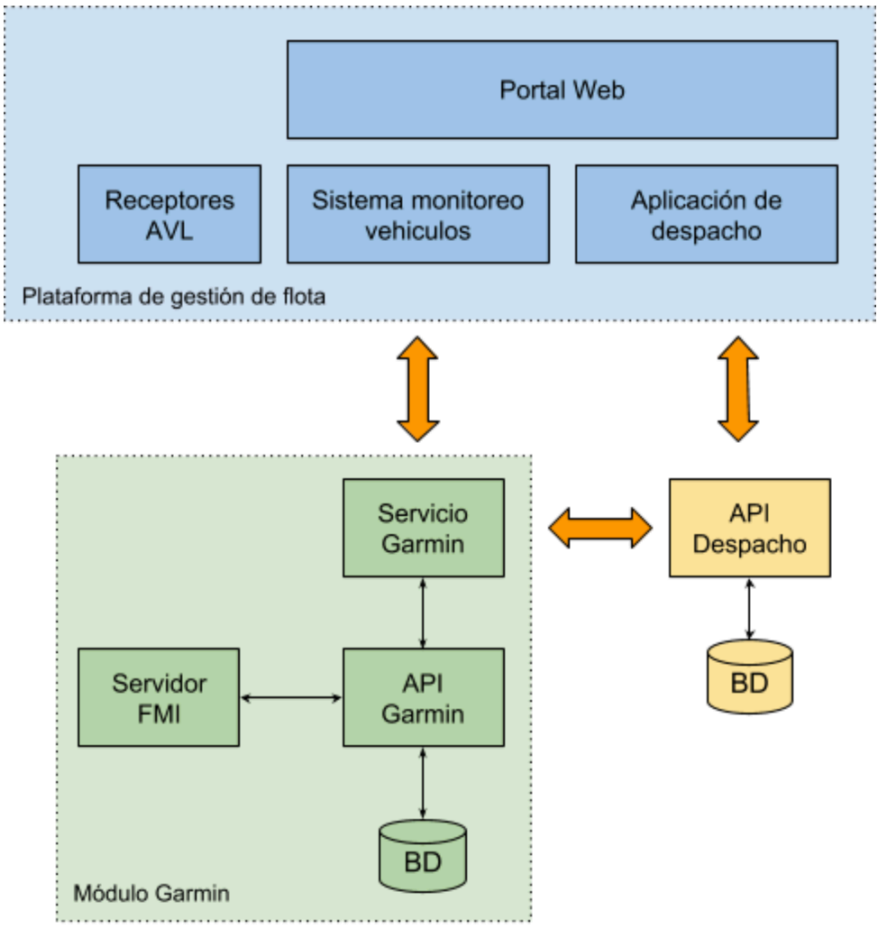

Figura 4.1: Arquitectura propuesta para el Backend de la solución. (Diagrama de autoría propia).

La arquitectura de alto nivel para que el Módulo Garmin se integre efectivamente al Backend se presenta en la figura 4.1. Los componentes que se señalan en la figura son:

- **Plataforma de gestión de flota:** Son aplicaciones o servicios existentes que harán uso de las funcionalidades expuestas por el Módulo Garmin.
- **Módulo Garmin:** Es el componente central del sistema a construir.
- **API despacho:** Corresponde a un servicio externo para el manejo de despachos que permite consultar puntos de detención. Su integración al sistema es opcional.

Las detenciones almacenadas en el sistema de despacho, pueden ser programados con la antelación que el cliente requiera. Por otro lado, su generación puede ser de realizada de forma manual o automática a través de algún software de optimización de rutas. Bajo esa premisa, el Módulo Garmin debe ser un cliente de ese servicio y ser capaz de leer dicha información para hacerla llegar oportunamente al conductor.

El punto de acceso al Módulo Garmin desde las aplicaciones externas se realiza a través del Servicio Garmin, el cual se comporta como un Gateway desde el punto de vista de la arquitectura. La interacción entre los componentes se define bajo invocaciones de servicios REST [14]. Las características más relevantes que impulsan este modelo de arquitectura son: Desempeño, Escalabilidad y Simplicidad de interfaces, entre otros atributos.

El diseño del flujo de información entre estos sistemas se discute en la sección 4.2. Por otro lado, la comunicación con los dispositivos embarcados se realiza con la ayuda del Servidor FMI, tal como se presentó en la sección 2.1.

4.2. Interacción de Módulo Garmin con aplicaciones externas

La API de despacho, a pesar de ser un componente externo al módulo de integración Garmin, es una pieza de software fundamental para nutrir al sistema de información relevante para el negocio.

Al simplificar aún más el diagrama de la figura 4.1, la arquitectura de los componentes del sistema se reduce a un esquema donde es más sencillo visualizar y analizar los flujos de información que se deben modelar (Ver figura 4.2).

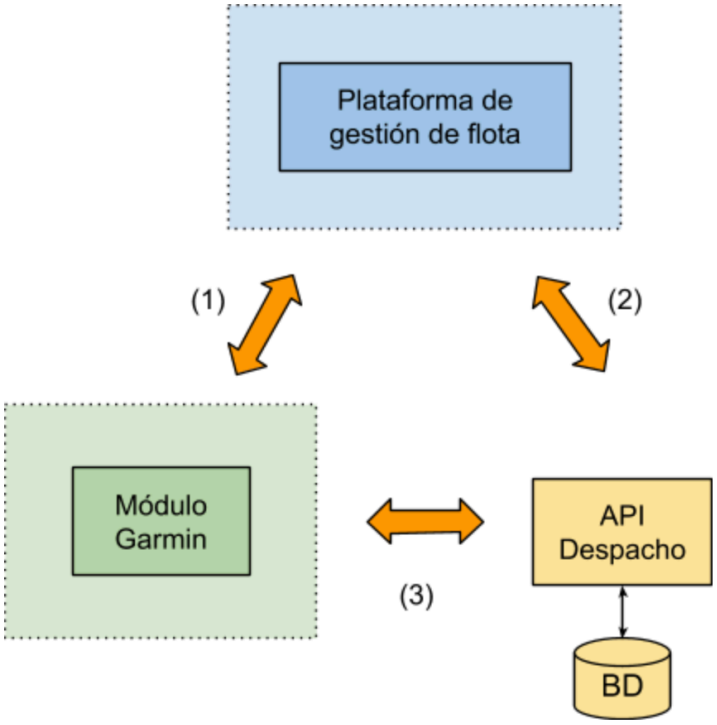

Figura 4.2: Diagrama simplificado de los componentes del sistema y los flujos de información entre ellos.

(Diagrama de autoría propia).

El flujo 1 de la figura 4.2 representa todos los eventos que ocurren entre la Plataforma de gestión de flota y el Módulo Garmin. De esta relación se desprenden las siguientes funcionalidades a implementar:

- Crear la asociación de un dispositivo Garmin con un determinado vehículo.
- Generar mensajes de texto hacia dispositivo embarcado.
- Consultar mensajes de texto respondidos por el conductor.
- Disparar la carga de paradas desde despacho.

En la misma figura 4.2, el flujo 2 representa la interacción entre la Aplicación de Despacho (parte del Sistema de gestión de flota) y su API de Backend respectiva. El objetivo es graficar que en este nivel se gestionan los puntos pertenecientes a itinerarios que eventualmente servirían para alimentar al Módulo Garmin. Las principales funciones que se consideran para el análisis:

- Administrar conductores, vehículos, clientes, paradas y despachadores.
- Crear órdenes de servicio asociadas clientes y sus respectivas paradas.
- Gestionar itinerarios que debe cumplir un conductor.
- Consultar estado de despachos.

El flujo 3 de la figura 4.2 representa la información que se comparte entre la API de Despacho y el Módulo Garmin. Las funcionalidades que están enfocadas en el envío de georreferencias al dispositivo Garmin son:

- Obtener puntos geográficos a visitar asociados a un determinado vehículo.
- Obtener métricas de cumplimiento para despachos definidos.
- Consultar datos complementarios del negocio como: órdenes de servicio o nombre de cliente.

Con el levantamiento funcional de los flujos antes mencionados, se puede especificar de manera más precisa la interfaz REST que debe proveer el Módulo Garmin.

4.3. Interacción de Módulo Garmin con equipos embarcados

Otra de las responsabilidades del Módulo Garmin es mantener la comunicación con el equipamiento embarcado. Para ello, este módulo debe considerar la implementación de un servidor que sepa interpretar el protocolo propietario FMI.

El Módulo Garmin debe convertir las instrucciones de alto nivel descritas en el apartado anterior en comandos propios de la tecnología Garmin. Esto se logra con la ayuda de una biblioteca de traducción FMI.

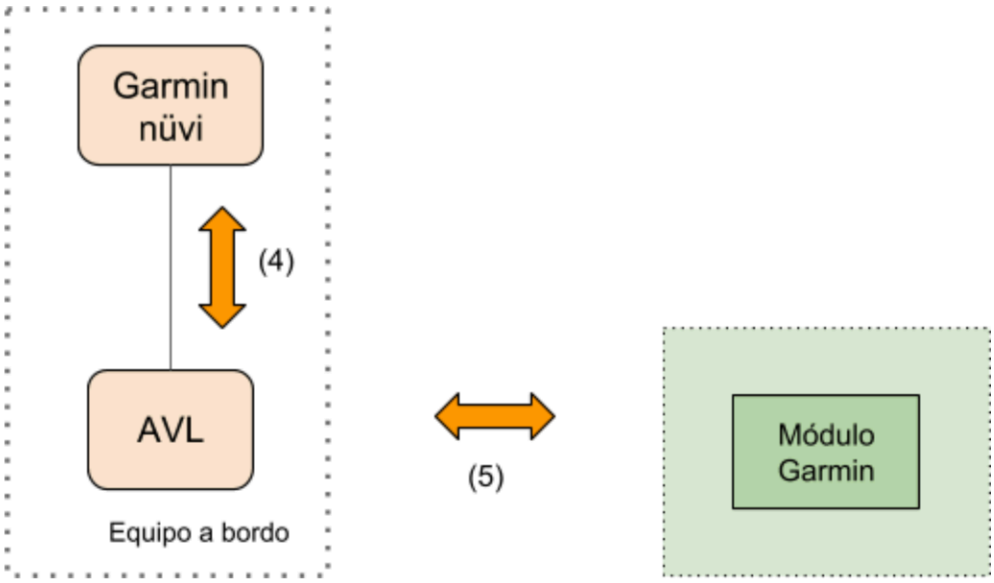

Figura 4.3: Flujos de información para equipo embarcado.
(Diagrama de autoría propia).

Las funcionalidades a bajo nivel que se deben considerar para el diseño de esta parte del sistema son:

- Administrar datos de paradas o detenciones.
- Enviar y recibir mensajes de texto.

De acuerdo al diagrama presentado en la figura 4.3, el flujo 4 corresponde a los eventos que pasan por la conexión física entre el equipo Garmin y el dispositivo AVL. En este sentido, la mensajería corresponde a los datos binarios que son replicados en el puerto serial del equipo AVL, gracias al modo PAD.

El flujo 5 de la figura 4.3 representa la comunicación sostenida con el equipo AVL desde el Módulo Garmin, utilizando la capa TCP/IP que provee el dispositivo AVL. Cabe señalar que la conexión es iniciada por el dispositivo AVL, siempre y cuando existan las condiciones para crear un canal GPRS usando la red celular.

Debido a las posibles intermitencias en la conectividad con la red celular, se deben tener en cuenta las algunas consideraciones durante la construcción de esta parte del sistema:

- Proveer una buena estrategia del manejo de la comunicación con los dispositivos ante la concurrencia de mensajes.
- Mantener actualizado el estatus de conexión TCP/IP de cada dispositivo.
- Gestionar la mensajería de sistema con equipo embarcado utilizando acuses de recibo.

4.4. Diseño del Módulo Garmin

En esta sección se describe el diseño conceptual del Módulo Garmin, el cual conforma el núcleo del sistema de integración con el equipo embarcado.

De acuerdo al diseño propuesto, la figura 4.4 muestra los elementos de software que conforman el Módulo de integración Garmin. Cada uno de estos elementos se describen a continuación:

- **API Garmin:** Interfaz de software que provee el acceso a todo el modelo de datos del módulo Garmin a través de métodos REST.
- **Base de datos:** Sistema de base de datos relacional que maneja la persistencia de los datos de la API Garmin.
- **Servidor FMI:** Encargado de crear un canal de comunicación TCP/IP con el equipamiento a bordo. Además este componente incorpora las bibliotecas de traducción del protocolo FMI, con las cuales se codifican y decodifican los mensajes entre el equipo de navegación Garmin y las aplicaciones de usuario.

- **Servicio Garmin:** Provee un punto de acceso de más alto nivel a las aplicaciones externas, permitiendo gestionar los atributos de los dispositivos y otros objetos del negocio, tales como vehículos y conductores.

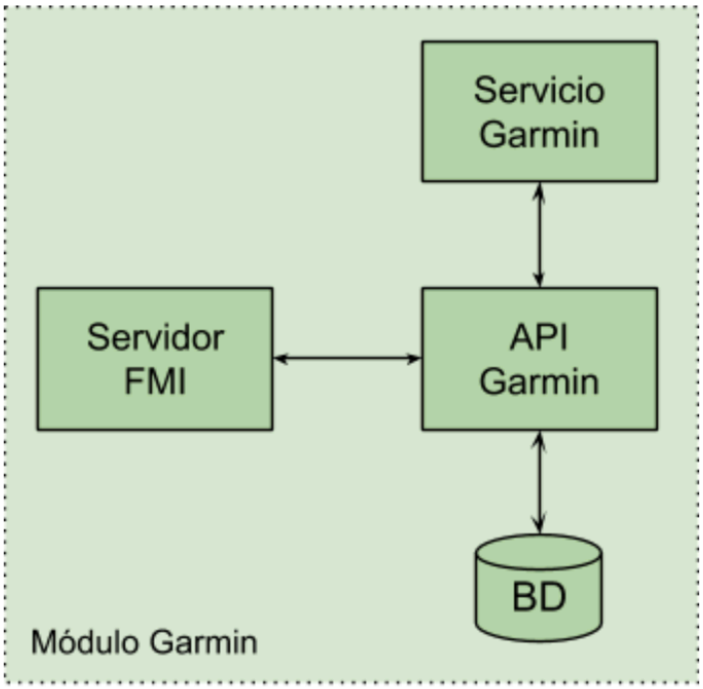

Figura 4.4: Módulo de integración Garmin y sus componentes.

La lógica de persistencia tanto de la mensajería como de los diccionarios del sistema se encuentran centralizados en la API Garmin. Toda la interacción entre el Servidor FMI y el Servicio Garmin se canaliza utilizando esta API (Ver figura 4.4). La estrategia antes descrita permite que exista solo un punto autorizado con acceso a la Base de datos relacional de este módulo de integración. Los otros componentes internos sólo acceden a las entidades a través de llamados REST hacia esta API.

A continuación se analizan dos posibles escenarios para la generación de un mensaje de texto en el sistema, de acuerdo con su origen:

- Aplicación externa
- Dispositivo embarcado

El estudio de ambos escenarios permite obtener los objetos y métodos que se requieren para completar el diseño de los llamados REST.

4.4.1. Generación de mensaje desde aplicación externa

En el contexto de este trabajo, la aplicación externa corresponde a la interfaz del sistema utilizada por el usuario Despachador.

La primera parte del flujo de la figura 4.5 consiste en hacer llegar el mensaje generado por una aplicación externa al repositorio de datos. Las operaciones de alto nivel que se pueden identificar son las siguientes:

- El usuario Despachador selecciona el equipo al cual se envía el mensaje e ingresa el texto (1).
- La aplicación externa (por ejemplo, App Web) completa los datos faltantes y se comunica con el servicio Garmin (2).
- El servicio Garmin valida a través de la API los datos ingresados (3).
- El servicio Garmin utiliza la API para almacenar el mensaje en la bandeja de mensajes y marca el status del mensaje como por enviar (4).
- Finalmente los componentes se van notificando de vuelta el resultado de la operación hasta llegar al usuario (5 - 6 - 7 - 8).

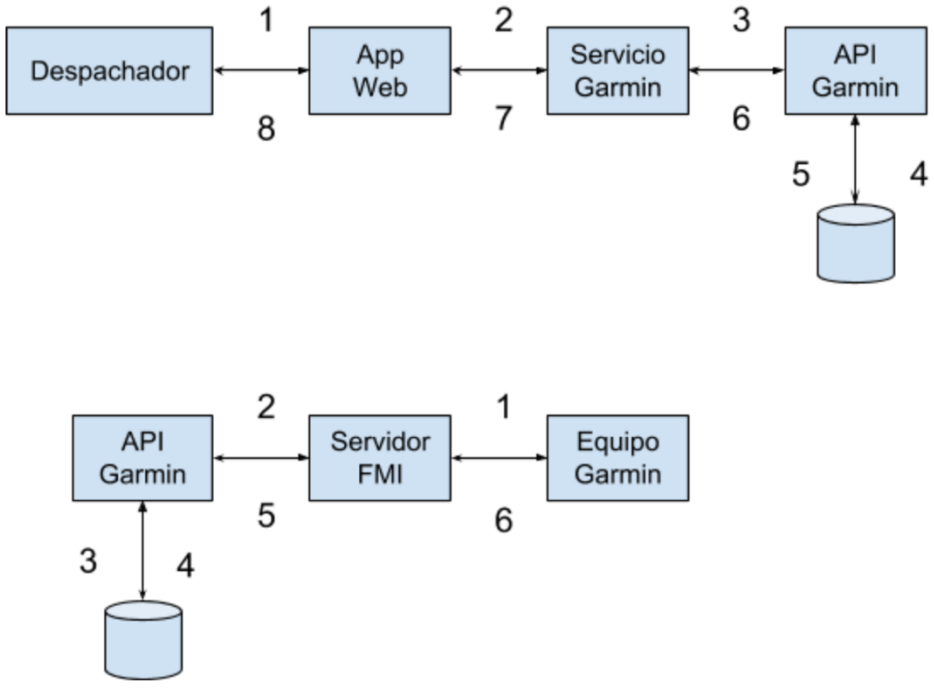

Figura 4.5: Secuencia operaciones para mensaje generado por aplicación externa. (Diagrama de autoría propia).

La segunda parte de la secuencia de la figura 4.5, consiste en rescatar el mensaje desde el repositorio de datos:

- El dispositivo Garmin, cuando se sincronice con el Servidor FMI, recupera el mensaje y actualiza el status como entregado (1 - 2 - 3).
- El usuario del dispositivo es notificado del mensaje por la alerta sonora y visual en el equipo (4 - 5 - 6).

4.4.2. Generación de mensaje desde un dispositivo Garmin

En este caso, la primera parte de la secuencia de la figura 4.6 la genera el usuario del dispositivo Garmin:

- El Conductor escribe un mensaje y queda internamente en el equipo en estatus por enviar (1).
- Cuando el dispositivo se sincronice con el Servidor FMI, se deja el mensaje en la bandeja del sistema (2 - 3 - 4).
- Se notifica al usuario que el mensaje ha sido enviado (5 - 6 - 7 - 8).

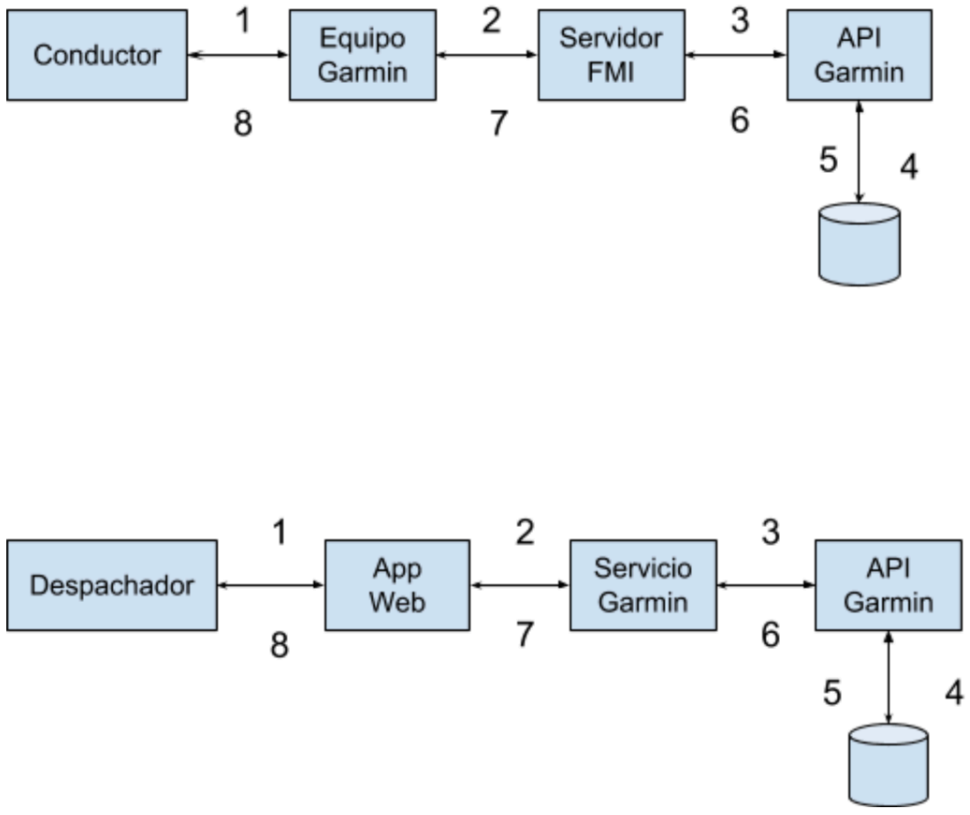

Figura 4.6: Secuencia de operaciones para un mensaje generado desde dispositivo embarcado.

(Diagrama de autoría propia).

Por otro lado, la segunda parte de la secuencia mostrada en la figura 4.6 se inicia con el usuario Despachador:

- El Despachador, a través de la aplicación externa (por ejemplo, sistema web) consulta al Servicio Garmin si tiene mensajes no leídos desde determinado dispositivo (1 - 2).
- El servicio Garmin reenvía la consulta a la API para obtener los mensajes respectivos (3 - 4).
- Se envía la respuesta al usuario que inició la consulta (5 - 6 -7 - 8).

4.5. Diseño de modelo de datos

La API Garmin es el componente del Módulo Garmin que permite elevar el nivel de abstracción del modelo de datos exponiendo las entidades de negocio a través de consulta a servicios REST. Para la construcción de este módulo se utiliza tecnología Java EE [15], donde además se hace uso del "framework" de desarrollo Spring [16] y de la biblioteca Jackson para producir objetos serializados en formato JSON [17].

De acuerdo con la arquitectura de software interna para la API Garmin representada en la figura 4.7, existen diversas capas que realizan la tarea de generar el desacople de la tecnología de acceso propia de la BD para los sistemas que son clientes de la información almacenada ahí (Servidor FMI y Servicio Garmin).

En el nivel más alto del diagrama están representadas las tecnologías relacionadas con el mundo Java EE, tales como Spring MVC, JPA y JDBC.

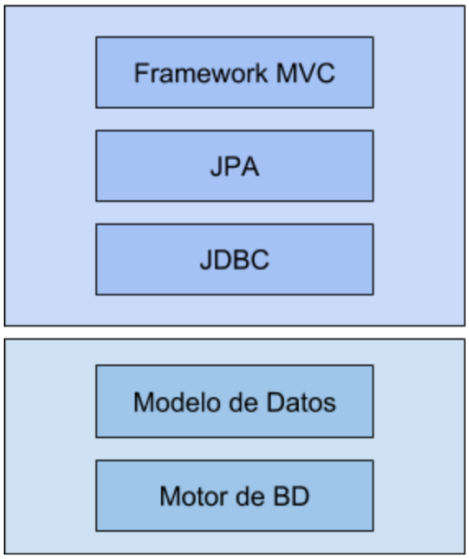

Figura 4.7: Capas de software para API Garmin.
(Diagrama de autoría propia).

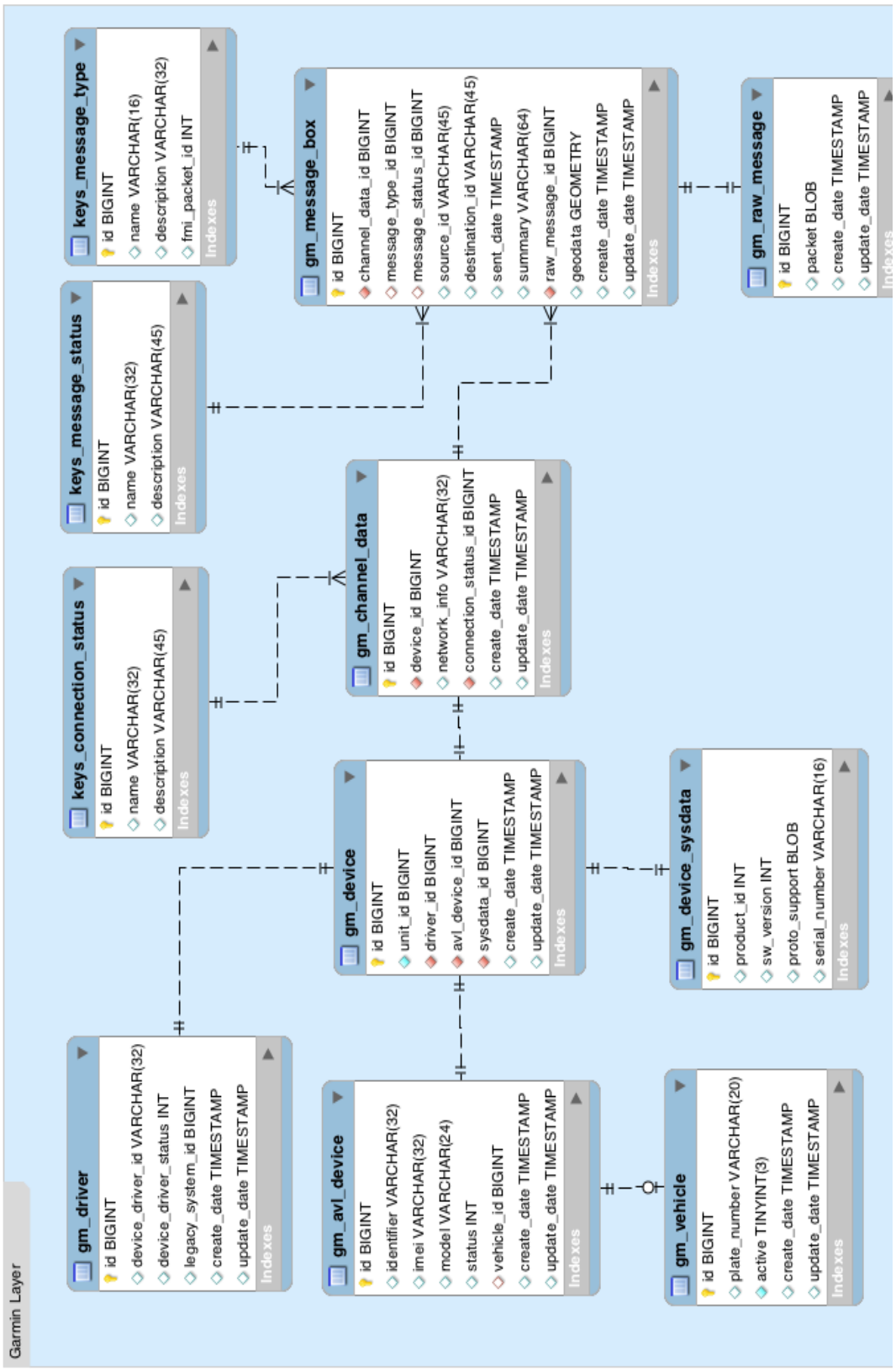

Figura 4.8: Modelo de datos para API Garmin.
(Diagrama de autoría propia).

Descendiendo en el diagrama de la figura 4.7, se puede apreciar el sistema de base de datos, representado lógicamente por el motor y el modelo de datos.

El diseño del modelo de datos se expone en la figura 4.8. Las tablas y sus atributos representan las entidades que necesita el Módulo Garmin para trabajar en su conjunto.

Tomando como referencia el modelo de datos presentado, se diseñan los objetos base de la API Garmin. Las clases que se generan a partir de esta estrategia de diseño se les conoce genéricamente como POJOs [18].

4.6. Diseño de métodos REST en API Garmin

Los métodos REST de la API son consultas y operaciones a realizar sobre las entidades del sistema, encapsulados bajo el paradigma de integración con micro servicios. A continuación se analizan, a modo de ejemplo, los métodos definidos para la entidad "Device" que representa al equipo Garmin y las posibles respuestas a obtener.

4.6.1. getById(id): Buscar dispositivo por identificador

El método "getById" se utiliza para consultar los atributos de un determinado dispositivo Garmin realizando la búsqueda a través del identificador interno y auto generado del sistema para dicho elemento. Si la consulta realizada es exitosa, el API debe retornar el objeto. En caso de que la búsqueda no sea satisfactoria, se espera una respuesta con el estatus respectivo. En la tabla 4.1 se presenta un cuadro resumen de la definición del método.

GET	/hn_gserver_api/{version}/rest/device/id/1
Body	
Response	{ "id": 1, "unitId": 3858320694, "driver": { "id": 2, "legacySystemId" : 5000 }, "avlDevice": { "id": 2, "identifier": "124" }, "sysdata": { "id": 2 } }
Response 2	{ "statusId": 404, "message": "Not Found", "timeConsumed": "2016-12-02T03:05:50Z" }

Tabla 4.1: Definición de método REST "getById" para la entidad "Device".
(Tabla de autoría propia).

4.6.2. getCount(): Obtener la cantidad de dispositivos

El método “getCount” es útil para saber cuántos dispositivos Garmin son gestionados por el sistema. Su respuesta es el valor entero de la cantidad. La tabla 4.2 presenta el resumen de la definición.

GET	/hn_gserver_api/{version}/rest/device/count
Body	
Response	{ "count": 2 }

Tabla 4.2: Definición de método REST “getCount” para la entidad “Device”.
(Tabla de autoría propia).

4.6.3. save(obj): Almacenar un dispositivo

El método “save”, se utiliza para almacenar o actualizar un dispositivo en el sistema. Cuando el atributo “id” del objeto se encuentra nulo en el cuerpo del mensaje, la API retorna el valor correlativo autogenerado respectivo. En la tabla 4.3 se pueden apreciar algunos valores de ejemplo para este llamado.

POST	/hn_gserver_api/{version}/rest/device/save
Body	{ "id": null, "unitId": 3858321063, "driver": null, "avlDevice": null, "sysdata": null }
Response	{ "message": "Object Saved","timeConsumed": "2016-12-02T03:23:16Z", "id": 3 }

Tabla 4.3: Definición de método REST “save” para la entidad “Device”.
(Tabla de autoría propia).

4.6.4. delete(obj): Eliminar un dispositivo

El método “delete” permite borrar dispositivos Garmin del sistema. En el cuerpo del mensaje HTTP debe ir el objeto a eliminar. La tabla 4.4 presenta un ejemplo de este método.

DELETE	/hn_gserver_api/{version}/rest/device/delete
Body	{ "id": 3, "unitId": 3858321063, "driver": null, "avlDevice": null, "sysdata": null }
Response	{ "message": "Object Removed","timeConsumed": "2016-12-02T03:25:07Z", "id": 3 }
Response 2	{}

Tabla 4.4: Definición de método REST “delete” para la entidad “Device”.
(Tabla de autoría propia).

4.7. Especificación de tecnologías para Módulo Garmin

Los componentes del Módulo Garmin, que representan el núcleo de esta solución, están diseñados para trabajar con tecnología Java EE. También se utilizan otros elementos como: frameworks, bibliotecas y servidores provistos bajo la filosofía de software libre, por lo que no se requieren licencias de software propietarios.

Las tecnologías que se utilizan en este trabajo fueron escogidas de acuerdo a la experiencia previa que se tiene con ellas dentro de la compañía. En la tabla 4.5 se presentan éstas y en algunos casos relevantes, se menciona la versión utilizada.

Tecnología	Descripción
Java / Java JEE	Plataforma de programación para desarrollar y ejecutar software de aplicaciones en el lenguaje de programación Java. Permite utilizar arquitecturas de N capas distribuidas y se apoya ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones. La versión utilizada es la 1.8.
Spring Framework	Conjunto de bibliotecas que facilitan la configuración y administración de objetos Java a través de la técnica de inyección de dependencia. La versión utilizada es la 4.1.
MySQL [19]	Sistema de gestión de base de datos relacional. La versión que se utiliza es la 5.6.
Apache Tomcat [20]	Contenedor web con soporte de servlets y JSPs. La versión a que se utiliza es la 8.0.
Hibernate [21]	Es una biblioteca para Java que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación.
Hibernate spatial [22]	Es una biblioteca para Java que extiende las funcionalidades de Hibernate para el manejo de datos geográficos.
Ubuntu [23]	Distribución del sistema operativo GNU/Linux.

Tabla 4.5: Tecnologías que se utilizan en el núcleo de la solución.
(Tabla de autoría propia).

4.8. Especificación de interfaz de usuario

Anteriormente en este capítulo se ha presentado un conjunto de consideraciones con el objetivo de modelar el núcleo del sistema. De acuerdo a cómo se ha planteado la arquitectura de esta solución, en el futuro se puede utilizar cualquier aplicación de Frontend de la plataforma de gestión de flota para hacer uso del Módulo Garmin, que se adhiera a los métodos REST definidos. No obstante, esta sección se enfoca en la especificación de una aplicación web para ser usada durante la realización de la prueba piloto, la cual se puede tomar como base para desarrollar una herramienta más elaborada.

La “Aplicación Web” debe permitir al usuario Despachador, previamente autenticado en el sistema, utilizar las funciones de mensajería interactiva y el envío de detenciones.

Junto con lo anterior, además debe facilitar la gestión de las entidades de negocio, tales como:

- Vehículos
- Conductores
- AVLs
- Dispositivos Garmin

Desde el punto de vista del diagrama simplificado de los módulos de la solución (Figura 4.9), la interfaz web se comporta como un cliente de capa de presentación, por lo que toda la lógica del negocio debe ser invocada al Módulo Garmin vía consultas REST.

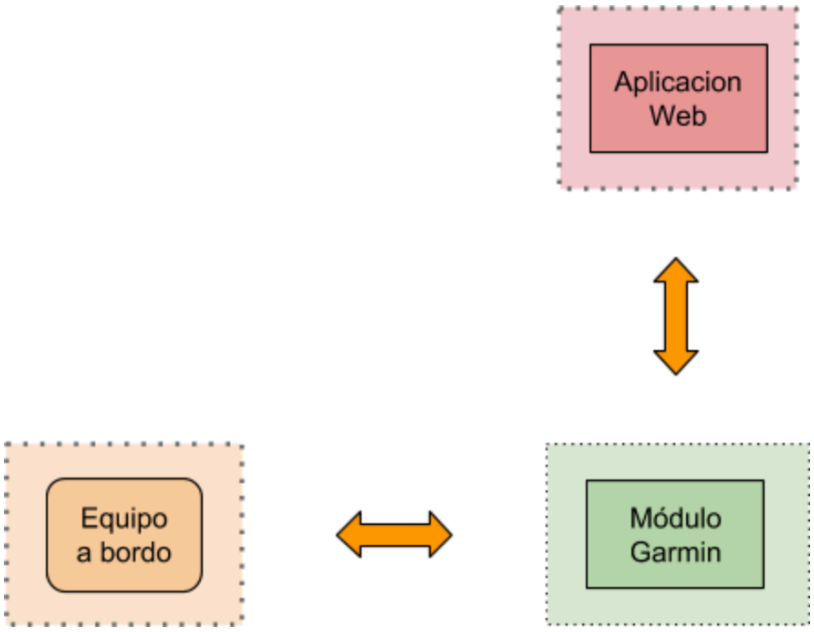

Figura 4.9: Diagrama simplificado de los módulos de la solución.
(Diagrama de autoría propia).

Para disponer de una mejor experiencia de usuario al utilizar el sistema, se propone el uso de la API Google Maps [24], esto permite externalizar la búsqueda y despliegue de la ubicación geográfica de los puntos de atención a un sistema GIS consolidado y ampliamente utilizado.

La interfaz debe considerar también un cuadro resumen tipo panel de control, para visualizar de manera efectiva la situación actual del sistema. La información que se puede desplegar en este panel considera:

- Indicadores de cumplimiento de las tareas de conducción.
- Últimos mensajes recibidos.
- Notificaciones del sistema.

La tecnología a utilizar en esta parte del sistema se presenta en la tabla 4.6. Básicamente se trata de componentes para ejecutar lenguaje PHP [25], por el lado del servidor y bibliotecas Javascript [26] por el lado del navegador cliente.

Tecnología	Descripción
Apache HTTP [27]	Servidor web HTTP de código abierto, para plataformas Unix, Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1
PHP	Lenguaje de programación de propósito general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico.
Codeigniter [28]	Codeigniter es un framework para el desarrollo de aplicaciones en PHP que utiliza el paradigma MVC.
Bootstrap Framework [29]	Framework web para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como, extensiones de JavaScript opcionales adicionales.
JQuery [30]	Biblioteca multiplataforma de JavaScript, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web.
API Google Maps [24]	Biblioteca que permite utilizar los servicios cartográficos de Google.

Tabla 4.6: Tecnologías utilizadas en la Aplicación Web.
(Tabla de autoría propia).

4.9. Especificación de infraestructura para el sistema

En este trabajo se define que todos los componentes del sistema se ejecuten en un mismo ambiente operativo. Las aplicaciones base que se deben habilitar se enumeran a continuación:

- Servidor de Base de datos (MySQL)
- Máquina Virtual Java
- Contenedor de Aplicaciones (Tomcat)
- Servidor Web (Apache + PHP)

Las condiciones de ejecución estimadas para la prueba piloto, permiten suponer que la carga del sistema sea mínima. Por ejemplo, para la mensajería asociada al envío y recepción de eventos con sólo un dispositivo Garmin y un sólo usuario Web durante una jornada normal de trabajo se estima que sean razonable los siguientes valores:

- Mensajes de texto: 50
- Generación de puntos de visita: 10
- Recepción de puntos de visita: 10
- Notificación de cumplimiento de visitas: <puntos> x <cambio estados> = 30

Estos valores se pueden proyectar a un máximo de 4 personas que desempeñen el rol de Conductor y un máximo de 3 usuarios Despachadores para obtener una aproximación de los eventos a procesar por el lado del Servidor. Desde el lado de la Aplicación Web, se pueden proyectar además las consultas realizadas por el navegador para refrescar el estatus del sistema, así como también las operaciones de mantenimiento de las entidades de negocio.

En base a criterio experto y basado en los antecedentes antes descritos, los requerimientos mínimos y una proyección de un aumento de al menos unas 5 veces en la demanda de este sistema en el mediano plazo, permiten estimar los requisitos del sistema base tal como se presenta en la tabla 4.7.

	S.O.	**RAM**	**Cores**	**HD**	**Uso**
Sistema base	Ubuntu 16.04	2 GB	1	40 GB	Servidor Aplicaciones y BD

Tabla 4.7: Requisitos del ambiente de ejecución.
(Tabla de autoría propia).

En cuanto a los servicios adicionales a ser considerados en el ambiente de pruebas, aparecen nuevos atributos de calidad para el sistema:

- Sincronización de hora con servidores NTP (Confiabilidad).
- Respaldo de configuraciones y sistema de archivos (Seguridad).
- Respaldo diario de base de datos (Seguridad).

En la medida que la criticidad del sistema lo amerite, se puede incorporar monitoreo remoto basado en los protocolo SNMP para los recursos del sistema y JMX para las aplicaciones Java.

5. Desarrollo e implementación

La solución analizada hasta ahora, se compone en definitiva por una Aplicación Web orientada al usuario Despachador, que hace uso de los servicios del Módulo Garmin para comunicarse con el equipo a bordo, y vice versa. El diagrama de la figura 5.1 representa una vista global de los componentes involucrados.

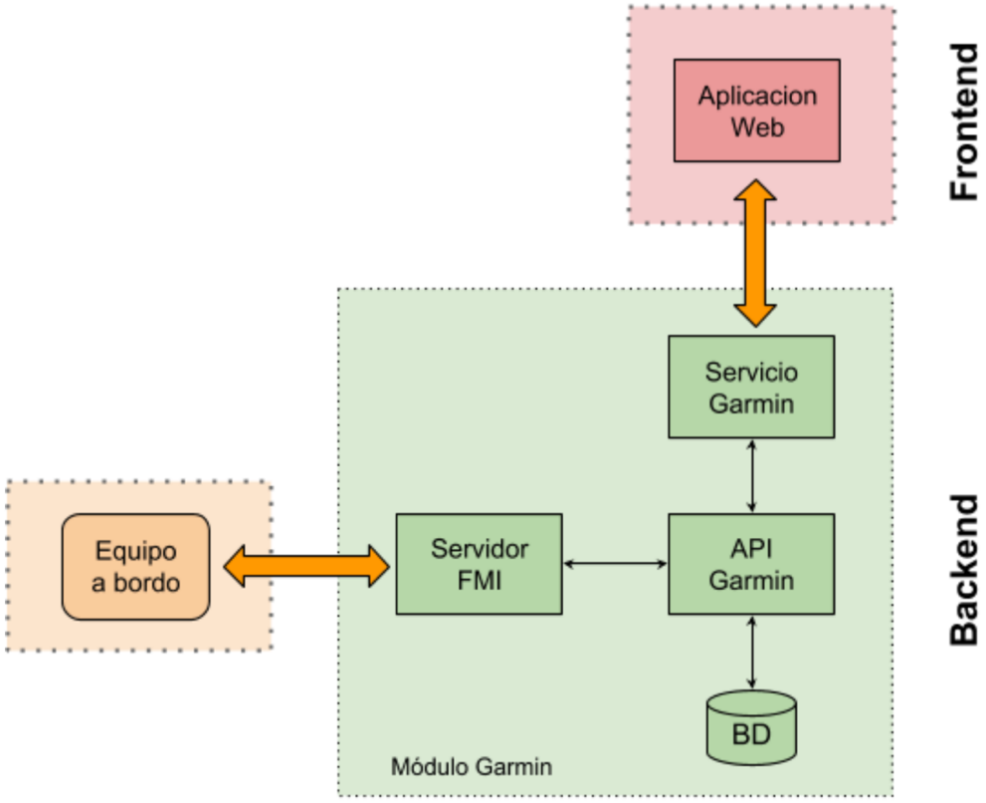

Figura 5.1: Vista global de los componentes del sistema.
(Imagen de autoría propia)

En este capítulo se aborda la construcción de la solución desde el punto de vista de los servicios de Backend y la aplicación de Frontend, con el objetivo de presentar aspectos particulares del proceso de desarrollo así como también señalar las consideraciones tomadas durante las pruebas unitarias para la implementación en el ambiente piloto.

5.1. Construcción e instalación de módulos de Backend

De acuerdo a lo especificado en la fase de diseño de este proyecto, existen 3 módulos que permiten cubrir las funcionalidades del Backend del sistema, estos son:

- API Garmin
- Servidor FMI
- Servicio Garmin

La construcción de estos módulos se ha realizado utilizando lenguaje Java con la ayuda del entorno de desarrollo Eclipse. El proceso de construcción de cada uno de ellos es similar y consiste en generar un proyecto maven [28] basado en un template "maven-archetype-webapp". Esto permite básicamente que la dependencia de las bibliotecas que utiliza cada uno de los módulos se puedan resolver de forma automática y que la estructura del proyecto Java sea más ordenada.

En la tabla 5.1 se presenta un resumen de los módulos y sus versiones. La instalación de "API Garmin" y "Servicio Garmin" se realiza sobre el servidor Tomcat 8.0, ya que utilizan funcionalidades del Framework Spring. Al momento de construir el archivo ejecutable para ambos servicios, éstos deben generarse como archivo WAR (Web Application Archive).

Nombre	Módulo	Versión	Descripción
API Garmin	hn_gserver_api	1.4	Contiene las funcionalidades para consultar y almacenar los objetos del núcleo de la aplicación.
Servidor FMI	hn_gserver_fmi	1.0	Habilita la conectividad con dispositivos AVL en modo PAD para manejar los eventos del navegador satelital nüvi.
Servicio Garmin	hn_gserver_service	1.4	Alimenta el módulo Garmin desde aplicaciones externas.

Tabla 5.1: Módulos de software del sistema.
(Tabla de autoría propia).

Por otro lado, el módulo "Servidor FMI" se ejecuta fuera del contexto de Tomcat, como una aplicación autónoma. Para su empaquetamiento se utilizó el formato JAR (Java ARchive). Más detalle de este componente se puede ver en Anexo E.

5.2. Pruebas de implementación para API Garmin

Las pruebas sobre la API Garmin consisten en una serie de consultas de tipo REST para comprobar su ejecución de forma correcta. Para ejecutar estas pruebas se hace uso de la herramienta Postman [29].

La figura 5.2 muestra parte de los métodos REST configurados en la interfaz de Postman para probar la API. Para cada objeto del sistema se crea una carpeta con el conjunto de consultas posibles.

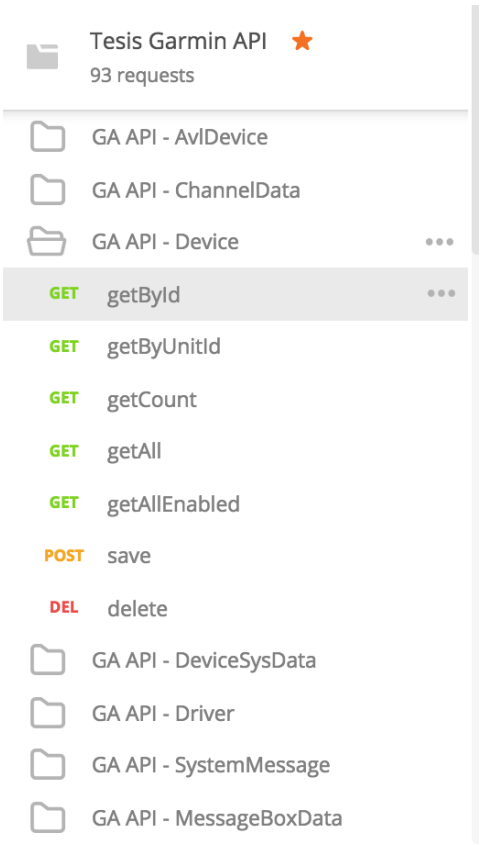

Figura 5.2: Configuración de métodos REST en Postman.
(Imagen de autoría propia).

Los problemas que se dan con mayor frecuencia durante las pruebas de integración, dicen relación con el acceso al recurso, ya sea por qué la URL no se ingresó de forma correcta o debido a la selección equivocada del operador http (GET o POST). Dependiendo de las bibliotecas que se utilicen para serializar/de-serializar los objetos, también es necesario definir atributos de la cabecera HTTP. En este caso “Content-Type” y “Accept” se han definido como “application/json”.

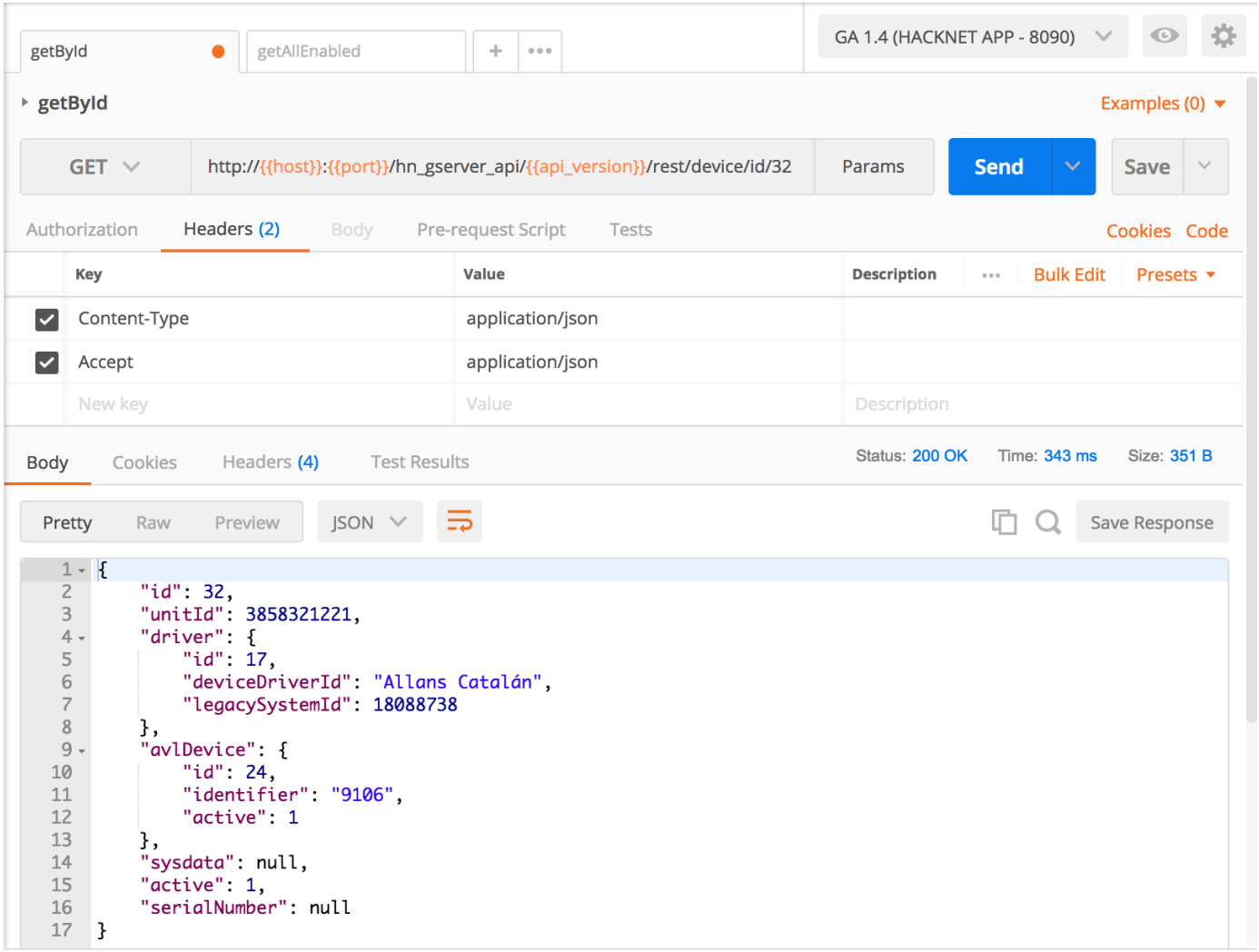

Figura 5.3: Pruebas unitarias de API Garmin.
(Imagen de autoría propia).

En la figura 5.3 se observa la consulta a la API de un dispositivo Garmin con identificador “32” y su respectiva respuesta. El método REST llamado es “getById()”. La respuesta correcta de este método retorna un Status 200 del protocolo HTTP, además del objeto en cuestión.

5.3. Pruebas implementación para Servicio Garmin

Dado que este servicio se modeló para ser el punto de entrada a las aplicaciones externas, las consultas realizadas a éste se propagan a la API Garmin, pero el formato de los objetos puede variar. En particular esto aplica cuando se trabaja con objetos que poseen coordenadas, ya que la biblioteca Java tiene una implementación distinta al PHP (Ver figura 5.4).

Java


```
{
    "id": 78,
    "unitId": 3858321221,
    "uniqueId": 6,
    "stopStatus": null,
    "stopIndexInList": null,
    "text": "Buses Lampa",
    "geodata": {
        "type": "Point",
        "coordinates": [
            -70.87126627116402,
            -33.278114332206144
        ]
    },
    "publishedDate": null
}
```


PHP


```
{
    "id": 78,
    "unitId": 3858321221,
    "uniqueId": 6,
    "stopStatus": null,
    "stopStatusText": null,
    "stopIndexInList": null,
    "text": "Buses Lampa",
    "latitude": -33.278114332206144,
    "longitude": -70.87126627116402
}
```


Figura 5.4: Comparación de objeto “Stop” para API vs Servicio Garmin.
(Imagen de autoría propia).

Al igual que la API Garmin, el Servicio Garmin se debe validar realizando consultas del tipo RESTful utilizando la herramienta Postman. En la figura 5.5 se puede apreciar la consulta “getById()” para el punto de detención “78”.

Los posibles puntos de falla de este módulo se pueden generar producto de errores en la comunicación con la API. Para el sistema piloto ambos módulos se alojan en el mismo servidor.

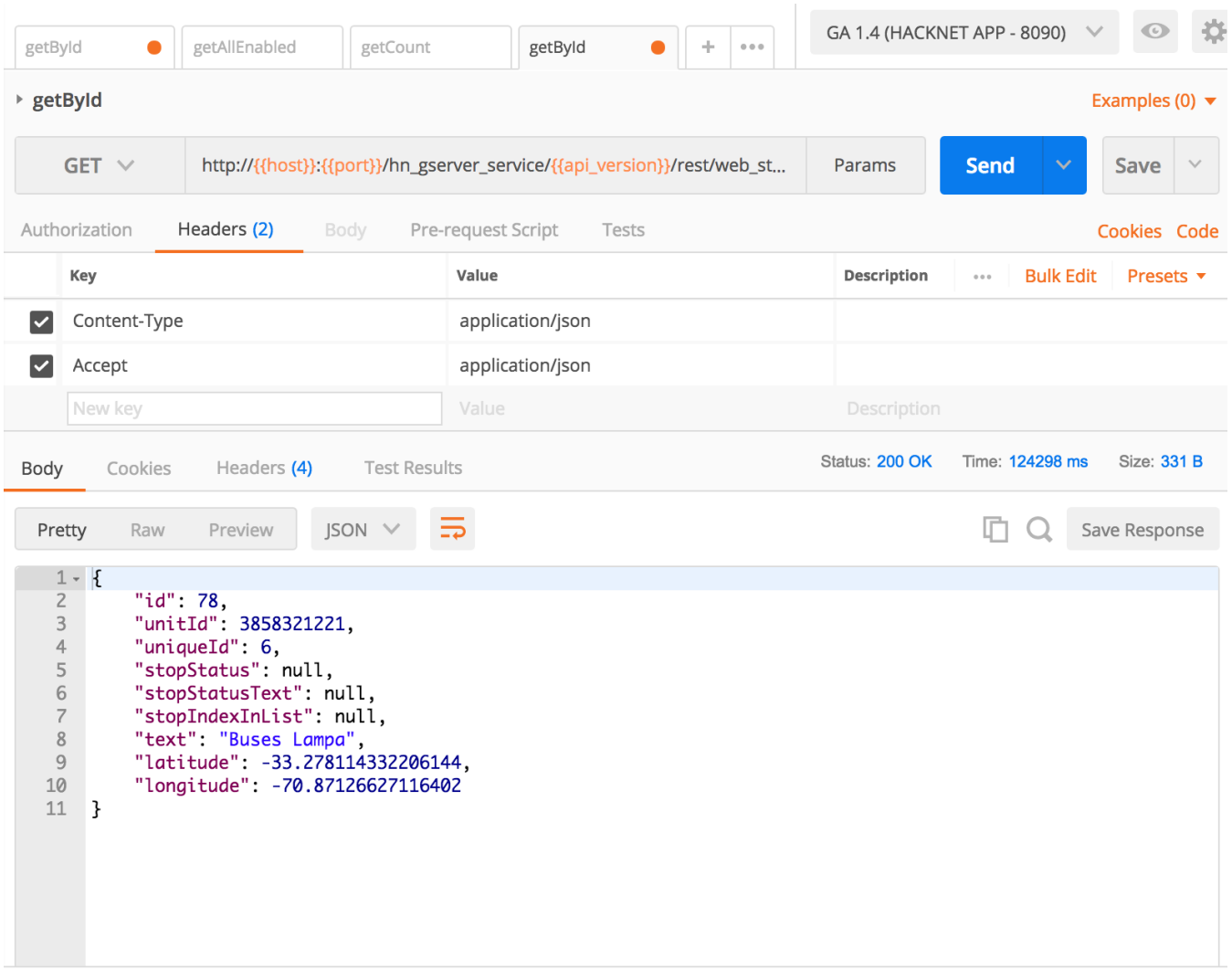

Figura 5.5: Pruebas unitarias de Servicio Garmin.
(Imagen de autoría propia).

5.4. Pruebas de implementación con equipos embarcados

Estas pruebas están enfocadas en validar la conectividad entre el equipamiento instalado a bordo y el servidor Garmin. Las pruebas que se realizan son las siguientes:

- Conexión de AVL al servidor
- Conexión de dispositivo Garmin al servidor a través de equipo AVL

5.4.1. Conexión de AVL al servidor

El equipo AVL utilizado es un Skypatrol TT8750. Una vez configurado en modo PAD permite utilizar el puerto serial RS232 físico para comunicarse con un servidor remoto transparentemente a través de un canal TCP/IP. La validación de esta funcionalidad se puede realizar revisando los archivos de log del módulo "Servidor FMI", tal como se muestra en la figura 5.6.

Lamentablemente el modo PAD del equipo no genera ningún identificador de dispositivo con el cual comprobar si efectivamente se trata del aparato en cuestión, por lo que esta prueba debe realizarse de manera unitaria.

```
2017-09-10 16:39:40,945  INFO WorkerRunnable - Iniciando WorkerRunnable
2017-09-10 16:39:40,960  INFO RestManager - api.host: 127.0.0.1,
api.port: 8090, api.name: hn_gserver_api, api.version: 1.3
2017-09-10 16:39:40,960  INFO WorkerRunnable - [17910004] Iniciando
initGarminProtocol
2017-09-10 16:39:40,960  INFO WorkerRunnable - [17910004] ALV conectado..
Sun Sep 10 16:39:40 CLST 2017
2017-09-10 16:39:40,960  INFO WorkerRunnable - [17910004] < Enviando
Enable Fleet Management
```

Figura 5.6: Log del Servidor FMI durante conexión AVL.
(Imagen de autoría propia).

Una mala configuración del modo PAD del AVL se presenta como la incidencia más crítica durante el desarrollo e implementación del Servidor FMI. Esto se manifiesta cuando al inspeccionar el archivo de log antes mencionado no se genera ningún intento de conexión, a pesar de que la red GPRS tiene buena cobertura.

5.4.2. Conexión de equipo Garmin al servidor a través de AVL

La conexión física del navegador Garmin con el puerto serial del equipo AVL se realiza con un cable específico provisto por el fabricante, llamado cable FMI. La prueba de conectividad consiste en revisar el archivo de log (Figura 5.7) del módulo "Servidor FMI" y detectar la secuencia de inicialización del protocolo Garmin.

```
2017-09-10 16:39:40,960  INFO WorkerRunnable - [17910004] < Enviando
Enable Fleet Management
2017-09-10 16:39:40,960 DEBUG FmiFactory - Creando
ENABLE_FLEET_MANAGEMENT_PROTOCOL_REQUEST [0x0000]
2017-09-10 16:39:40,960 DEBUG FmiFactory - FeaturesList: 05 00 01 80 02
80 0a 00 0b 00 0c 00
2017-09-10 16:39:40,960 DEBUG FmiFactory - appPayload
genEnableFleetManagementRequest: 10 a1 0e 00 00 05 00 01 80 02 80 0a 00
0b 00 0c 00 28 10 03
2017-09-10 16:39:45,439 DEBUG FmiParser - loadRawMessages - inputQueue
size: 2
2017-09-10 16:39:45,439 DEBUG FmiParser - [17910004] > ACK [0x06]
2017-09-10 16:39:45,439 DEBUG WorkerRunnable - [17910004] < Solicitando
Product Id And Support Data
2017-09-10 16:39:45,439 DEBUG FmiFactory - Creando
PRODUCT_ID_AND_SUPPORT_REQUEST [0x0001]
2017-09-10 16:39:45,439 DEBUG FmiFactory - appPayload
genProductIdAndSupportRequest: 10 a1 02 01 00 5c 10 03
2017-09-10 16:39:49,901 DEBUG FmiParser - loadRawMessages - inputQueue
```

Figura 5.7: Log del Servidor FMI durante conexión Garmin.
(Imagen de autoría propia).

Asumiendo que la conexión entre el AVL y el servidor es correcta, las malas condiciones del cable FMI es causal de falla en esta etapa. Durante la implementación piloto fue necesario reemplazar este componente en un vehículo por uno nuevo.

5.5. Construcción de Aplicación Web para Frontend

De acuerdo a lo mencionado en la etapa de diseño, se necesita contar con una Aplicación Web que permita gestionar los elementos del negocio y operar sobre las funcionalidades del Módulo Garmin. Esta aplicación cumple con el objetivo de ser el Frontend de la solución para la implementación piloto, la cual se ha construido usando el lenguaje PHP con apoyo de elementos en Javascript.

Luego de realizar la autenticación del usuario en el sistema, se presenta la pantalla inicial de la Aplicación Web (Figura 5.8), la cual corresponde a un panel de resumen de las variables más importantes de la operación. En la parte superior, bajo la barra de navegación, se muestra el estatus de cumplimiento de las tareas de conducción programadas. Estas están agrupadas en 4 categorías de viajes: “Completadas”, “Pendientes”, “Canceladas” e “Inactivas”. Al hacer clic a cada una de las cajas se puede ver el detalle de cuáles son estos viajes y a qué dispositivo de navegación asociado corresponde.

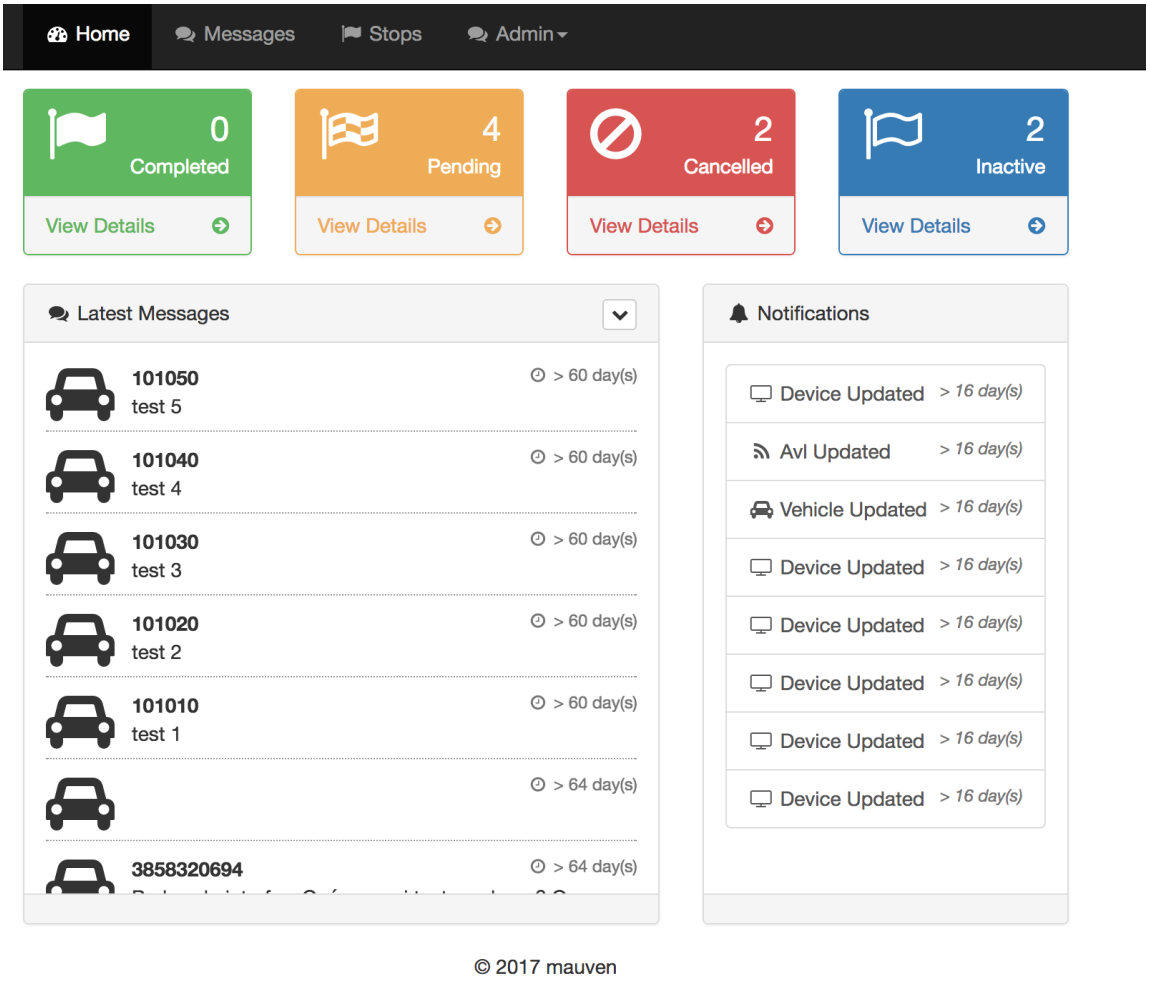

Figura 5.8: Panel de inicio de la Aplicación Web.
(Imagen de autoría propia).

El sub panel inferior izquierdo corresponde a los últimos mensajes de texto enviados por los conductores. El sub panel inferior derecho muestra las notificaciones del sistema, tales como creación de nuevos conductores, modificación de parámetros del vehículo, etc.

Avanzando en la barra de navegación, en la figura 5.9 se presenta la pestaña de “Mensajes”, la cual permite utilizar la funcionalidad de mensajería de texto. Una vez seleccionado el dispositivo con que se quiere comunicar, se despliega el historial de los mensajes. En la parte inferior aparece un cuadro para escribir el texto que se necesita a enviar.

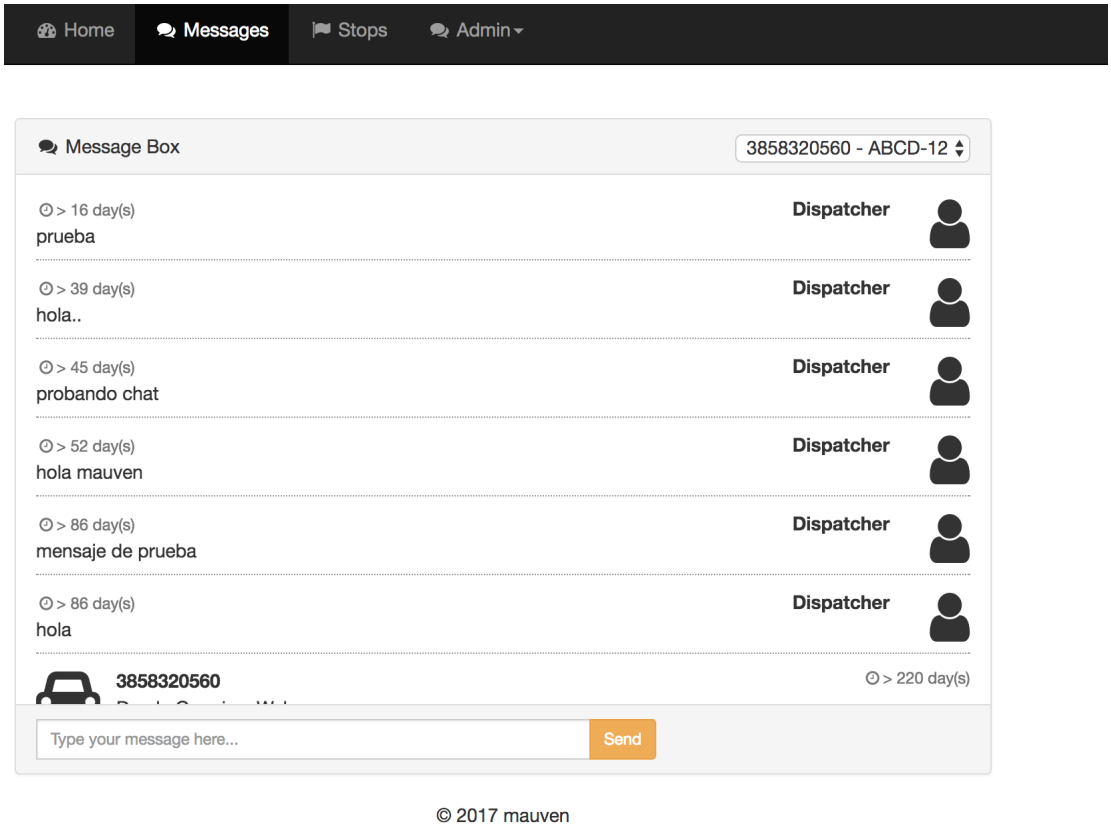

Figura 5.9: Mensajería de texto.
(Imagen de autoría propia).

El siguiente módulo de la barra de navegación corresponde a “Detenciones”, cuya imagen se puede apreciar en la figura 5.10. La funcionalidad de gestión de detenciones funciona de la siguiente manera: una vez seleccionado el móvil, aparece el listado de todas las paradas programadas por el usuario Despachador para un determinado vehículo. La creación de un nuevo punto de detención se realiza con el botón inferior del cuadro. Al hacer clic aparece una nueva ventana donde se deben ingresar los datos de georreferenciación o bien hacer la búsqueda a través de la dirección del destino.

Cada una de las detenciones tiene una bandera con distinta iconografía, para representar el estatus de ejecución del viaje. Además se puede ver una vista previa de la ubicación una vez que ya ha sido guardada (ver figura 5.11).

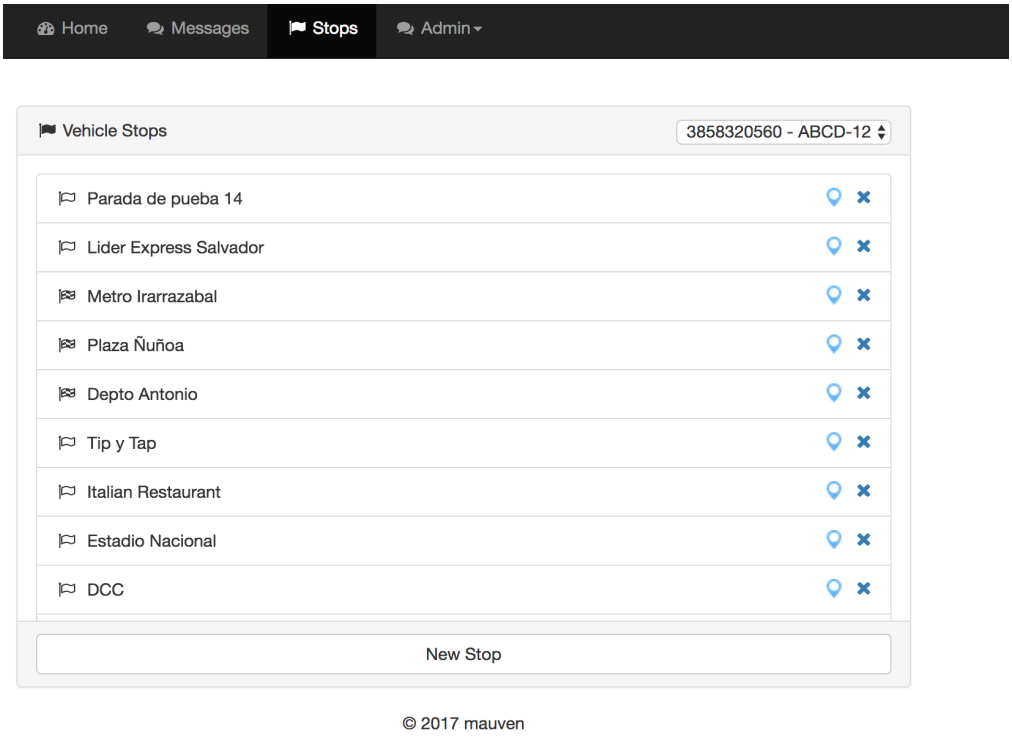

Figura 5.10: Administrador de detenciones.
(Imagen de autoría propia).

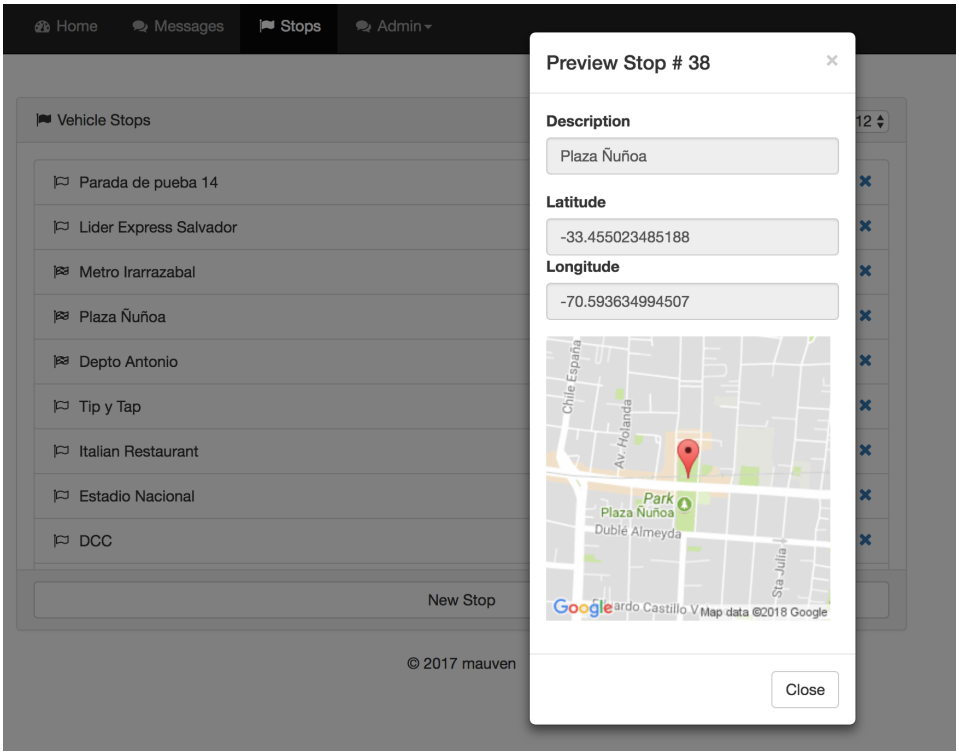

Figura 5.11: Vista previa de las detenciones creadas.
(Imagen de autoría propia).

La Aplicación Web también permite administrar las entidades: “Vehículos”, “Conductores”, “AVLs” y “Dispositivos Garmin”. En la figura 5.12 se presenta la vista del formulario de creación de vehículos. Los campos que se solicitan para generar un nuevo vehículo son: la patente, una descripción breve del vehículo y si éste se encuentra disponible o no para trabajar. Para el correcto funcionamiento del sistema, al menos es necesario indicar la patente del vehículo donde se encuentra instalado el equipo Garmin.

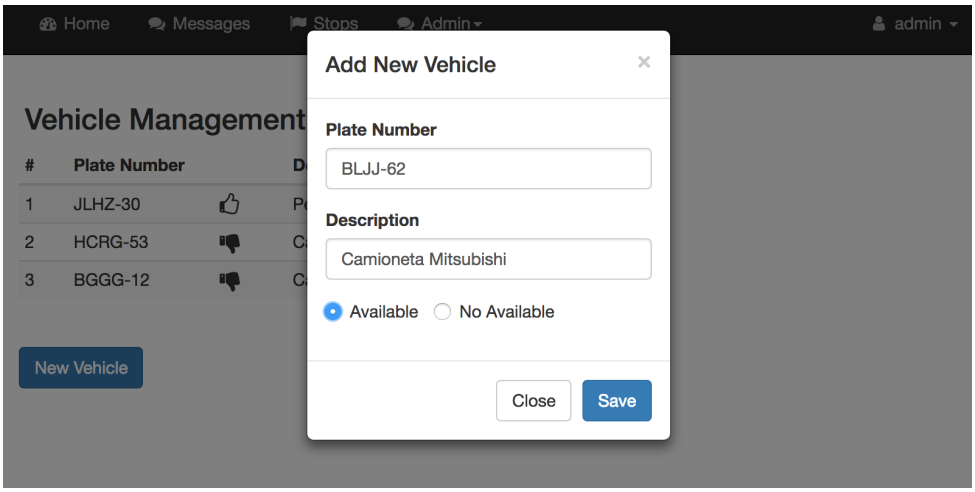

Figura 5.12: Creación de vehículo en Aplicación Web.
(Imagen de autoría propia).

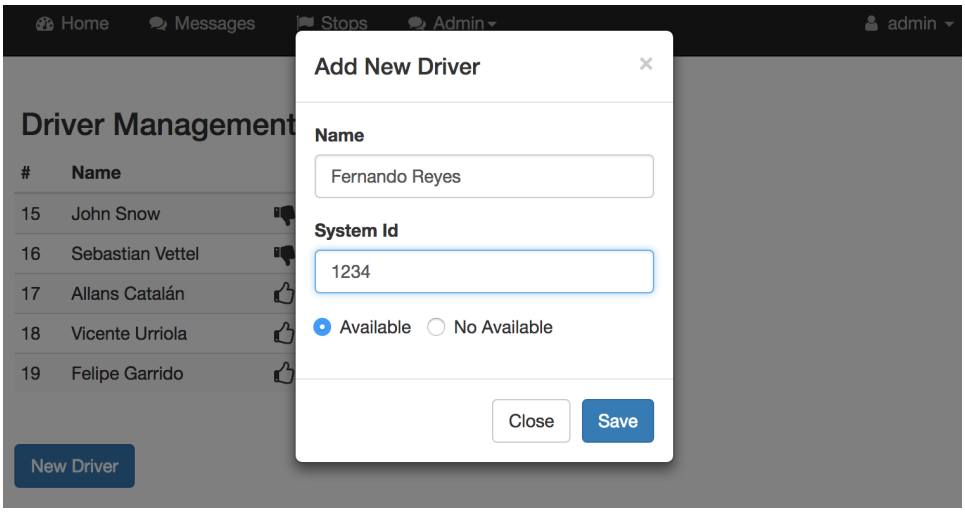

Figura 5.13: Creación de conductor en Aplicación Web.
(Imagen de autoría propia).

La administración de conductores se muestra en la figura 5.13 donde se observa el formulario de creación de un nuevo conductor en la aplicación. De forma análoga existen mantenedores para los dispositivos AVL y Garmin.

5.6. Pruebas de implementación Aplicación Web

La prueba consiste en que el Despachador utilice la Aplicación Web para enviar una parada conocida y que el conductor reciba la información en el dispositivo Garmin, para luego seguir hasta el destino. Antes de realizar la prueba, se comprueba que en el sistema existan los siguientes objetos del negocio:

- Vehículo
- Conductor
- Dispositivo AVL
- Navegador Garmin

El usuario Despachador crea una parada tal como se muestra en la figura 5.14. Luego de escribir la dirección del lugar de destino, la API de Google Maps completa la información de latitud y longitud respectiva. En caso de que la dirección no sea suficiente para especificar el punto de detención, la misma API de Google Maps permite desplazar el punto de forma manual.

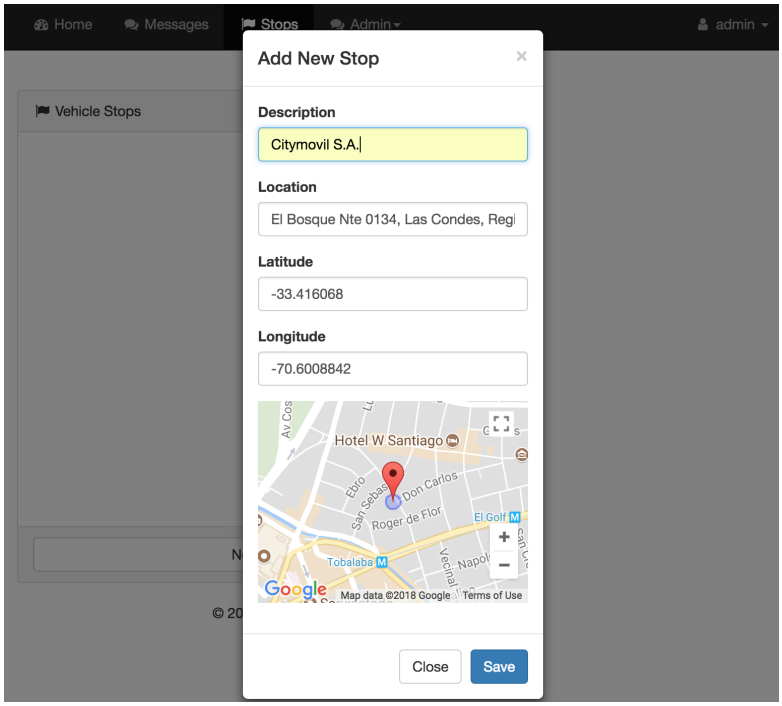

Figura 5.14: Creación de parada en sistema web.
(Imagen de autoría propia).

Al presionar el botón para guardar la detención, ésta se almacena en la BD hasta que el dispositivo Garmin se contacte con el sistema para rescatar la indicación. En la figura 5.15 se puede visualizar cómo se despliega la detención indicada en el equipo de navegación.

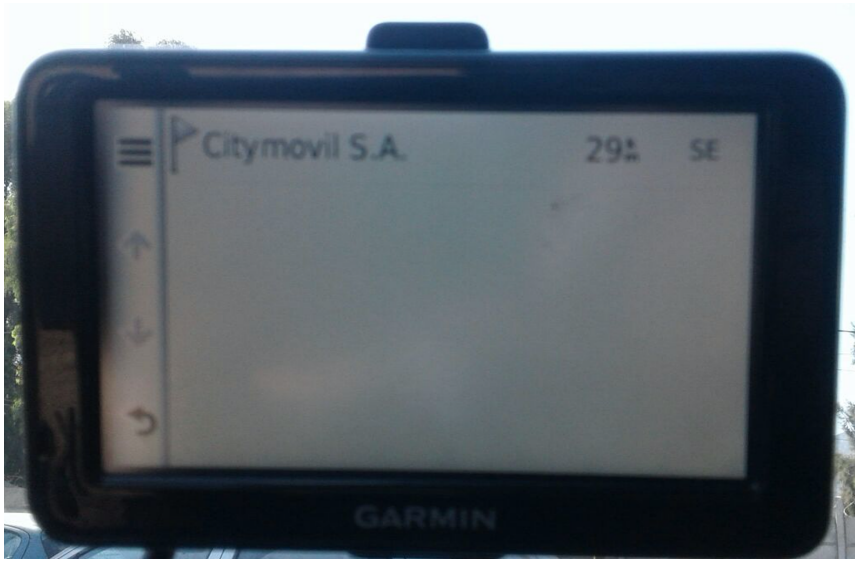

Figura 5.15: Visualización de parada en equipo Garmin.
(Diagrama de autoría propia).

Los problemas de cobertura con la red celular son los que tienen mayor incidencia directa en el fracaso de esta prueba.

6. Evaluación del Sistema

En este capítulo se describe el proceso para validar la solución desde el punto de vista de los usuarios, basándose en su experiencia con la implementación piloto. Las preguntas de la evaluación buscan obtener una correlación con los objetivos específicos de este trabajo y se enfocan en los siguientes aspectos:

- Evaluar procesos y herramientas para la tarea de gestionar visitas en terreno por parte de despachadores.
- Buscar una valoración de la eficiencia de la gestión de las órdenes de trabajo y los tiempos estimados de coordinación y ejecución de las visitas.
- Estudiar el cumplimiento de los servicios de despacho, desde el punto de vista de los administradores y con ello el impacto de la imagen de la empresa de cara al cliente.

Dado que se trata más bien de una evaluación de tipo sumativa, tal como se mencionó en la sección 3.5., se analizan aspectos que se puedan contrastar en dos instancias distintas del proceso de implementación.

Con respecto a la cantidad de personas que conforman el área de soporte y que su tarea está enfocada en los roles relacionados con las tareas de despacho, se puede mencionar que es reducida. De este grupo, se busca cubrir al menos los 3 roles necesarios para el análisis. Por temas de disponibilidad participan en definitiva 4 personas en el proceso de evaluación: 1 Administrador, 1 Despachador y 2 Conductores.

Una de las actividades desarrolladas con los usuarios como tarea previa a la evaluación, es la inducción al proceso y a la problemática a evaluar, con el objetivo de darles un mayor contexto al valorar los ítems consultados. La tarea siguiente, antes de comenzar la implementación piloto es realizar la evaluación inicial con el objetivo de generar una base comparativa. Con el procesamiento de los datos de cada encuesta, se obtienen los puntajes presentados en la tabla 6.1, para cada uno de los usuarios.

De la información obtenida de la tabla 6.1, uno de los análisis que se desprenden es comparar los niveles de percepción de cada usuario ante las preguntas generales. Para visualizar estos valores de mejor forma se construye el gráfico presentado en la figura 6.1.

Usuario	Puntaje a preguntas Generales	Puntaje a preguntas Específicas
Administrador	58	16
Despachador	61	7
Conductor 1	52	10
Conductor 2	55	7

Tabla 6.1: Resumen de Nivel de Satisfacción para la encuesta inicial.
(Tabla de autoría propia).

De acuerdo al gráfico de la figura 6.1, se evidencia que los usuarios que están en labores en terreno tienen una apreciación un poco más crítica del proceso de planificación y gestión de las visitas que quienes trabajan en labores de oficina, al menos antes de la ejecución del piloto. Este resultado global se correlaciona con las respuestas a la primera pregunta de la encuesta: *"¿Cuál es su valoración general de la planificación de visitas a terreno?"*, dónde Administrador y Despachador indican una buena percepción del proceso versus los Conductores que señalan una apreciación regular.

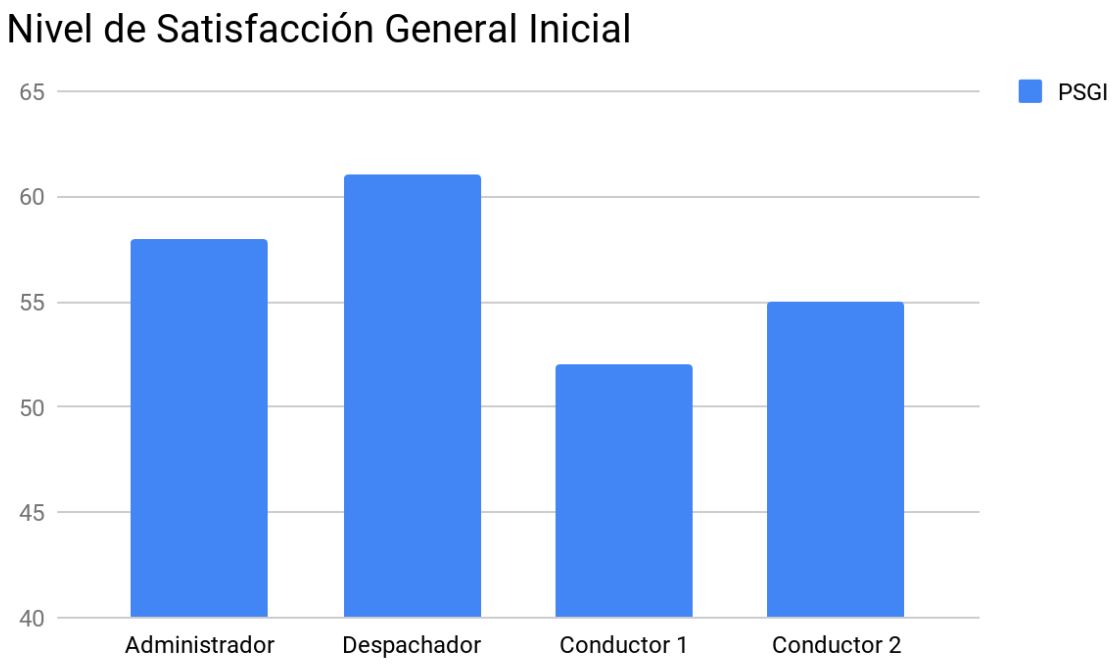

Figura 6.1: Gráfico de Nivel de Satisfacción inicial a preguntas generales.
(Imagen de autoría propia).

Otra situación interesante se presenta cuando se revisan las respuestas ante la pregunta *"¿Cómo evaluaría las herramientas informáticas que la empresa dispone actualmente para facilitar el desplazamiento a destino?"*, En este caso, las apreciaciones de todos los encuestados son disímiles, ya que no hay dentro del área un software formal y que todos utilicen.

Cuando se revisa la respuesta del usuario Despachador ante la pregunta *"¿La planificación se realiza con la anticipación adecuada?",* llama la atención que es el único que le da una valoración "siempre" versus los demás usuarios que indican "a veces".

La actividad siguiente que se aborda es capacitar a los usuarios directos en el uso del sistema y acompañarlos durante la ejecución del piloto, en la medida que fuese necesario.

En el inicio del piloto se constatan algunas situaciones que pueden incidir en la eficiencia de área, como por ejemplo que la tarea de planificación de viajes no es liderada por un rol definido, lo cual genera que cada usuario registre de manera libre y muchas veces informal la planificación del día. Bajo ese mismo escenario, se detectó que el canal de origen de solicitudes de visitas a terreno es diverso, ya que estas pueden ser requeridas a la mesa de ayuda vía email o por teléfono, pueden ser informadas por el área comercial o incluso por el soporte TI.

Otra situación anómala es que en varias oportunidades se observa que los conductores gestionan su itinerario de forma proactiva cada mañana, si es que no hay incidencias con mayor prioridad o solicitudes de clientes preferenciales, en vez de hacerlo el Despachador.

Con el final la implementación piloto, se realiza una nueva encuesta y luego de procesar estos datos, se generan los resultados presentados en la tabla 6.2.

Usuario	**Puntaje a preguntas Generales**	**Puntaje a preguntas Específicas**
Administrador	61	17
Despachador	64	8
Conductor 1	61	10
Conductor 2	58	8

Tabla 6.2: Resumen de Nivel de Satisfacción para la encuesta final.
(Tabla de autoría propia).

Al igual que en caso inicial, es útil generar un gráfico para comparar los niveles de percepción de cada usuario ante las preguntas generales, en base a los datos de la tabla 6.2. Este gráfico se presenta en la figura 6.2. A primera vista, las apreciaciones generales antes y después de la implementación parecieran ser similares.

En la figura 6.3 se presenta un gráfico comparativo entre el puntaje general de las tablas 6.1 y 6.2. En este gráfico se evidencia un incremento del nivel de percepción para todos los usuarios.

Un primer análisis que se deriva de la figura 6.3 es el incremento notable de la percepción del Conductor 1 con respecto a las preguntas generales de la primera medición. Esto puede tener su origen en una mayor participación de éste usuario dentro del proceso de despacho durante el piloto y a una utilización cada vez más activa del sistema.

En los demás usuarios, el aumento de su percepción tiene que ver principalmente con una mayor valoración de la planificación bajo el nuevo escenario de trabajo y en menor medida con la inclusión de herramientas informáticas en el área.

El análisis siguiente es evaluar en cuánto porcentaje aumentan los niveles de satisfacción para cada rol. Para este cálculo se debe considerar también el puntaje asociado a las preguntas específicas. En el caso de rol “Conductor”, se promedian los valores obtenidos para ambos encuestados. En la tabla 6.3 se presentan los valores porcentuales de aumento esperado (de acuerdo a los objetivos) y el incremento real (encuesta) para cada rol.

Usuario	**% incremento esperado**	**% incremento real**
Administrador	10	5
Despachador	20	6
Conductor	15	10

Tabla 6.3: Porcentajes de nivel satisfacción esperado v/s real por rol.
(Diagrama de autoría propia).

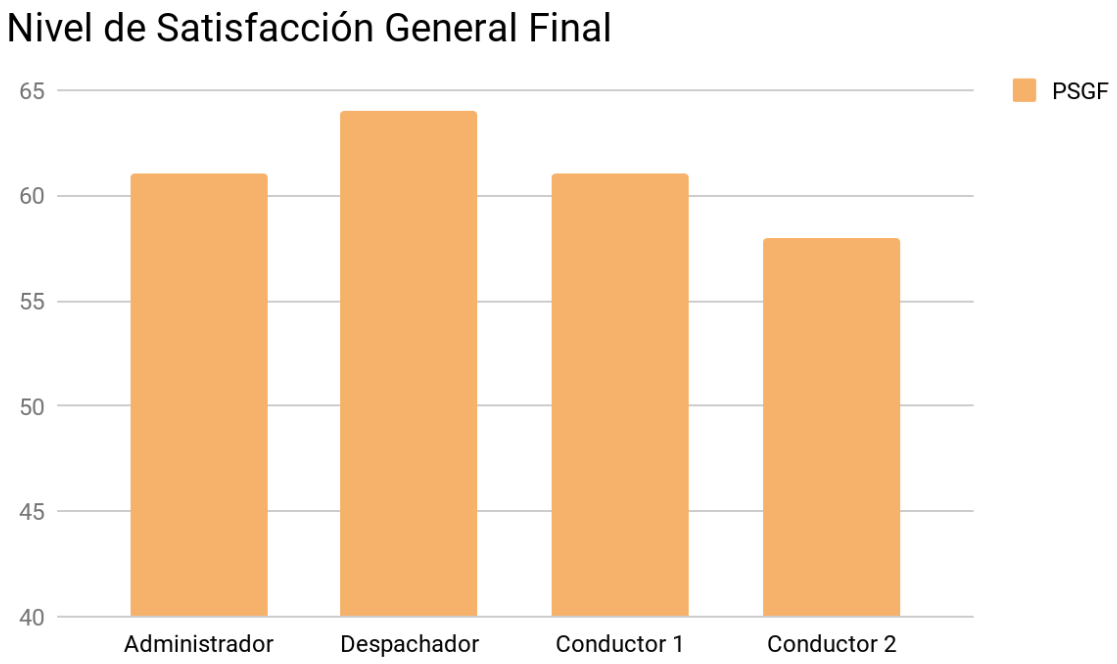

Figura 6.2: Gráfico de Nivel de Satisfacción Final a preguntas generales. (Imagen de autoría propia).

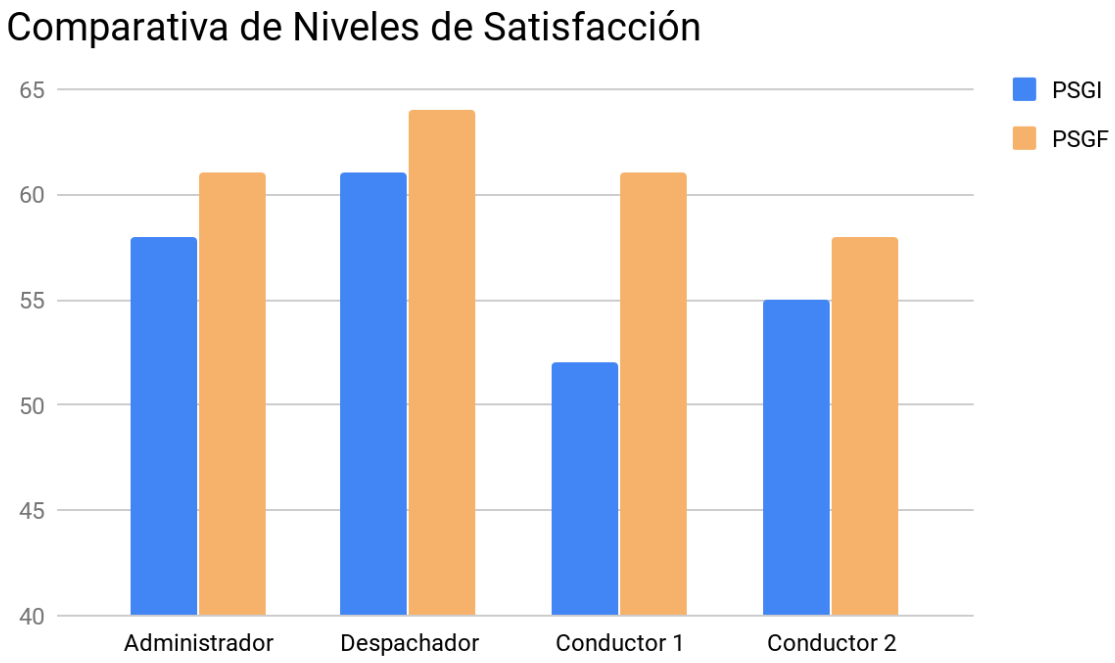

Figura 6.3: Niveles de percepción inicial y final a preguntas generales. (Imagen de autoría propia).

De la tabla 6.3 se desprende que el incremento en la satisfacción real no alcanza los valores esperados definidos en los objetivos específicos. De acuerdo con la tabla 6.4, el incremento en el nivel de satisfacción para el usuario Administrador fue cubierto de forma parcial, con un 50%. El nivel de satisfacción del Despachador sólo estuvo en un 30% de lo esperado. Por el lado de los Conductores, el nivel estuvo en un 66% de lo esperado (promedio). En este caso, el espacio de mejora de los niveles de satisfacción de Conductores aún puede crecer debido al menor puntaje global obtenido.

Usuario	**% cumplimiento objetivo**
Administrador	50
Despachador	30
Conductor	66

Tabla 6.4: Porcentaje de cumplimiento de los objetivos específicos de cada rol. (Diagrama de autoría propia).

En líneas generales, el cumplimiento parcial de los objetivos específicos se puede explicar debido a un bajo nivel de madurez del área y por ende a la informalidad en algunos de procesos que desarrollan relacionados con la gestión de visitas a terreno, lo que dificulta el obtener un mayor beneficio a intentos de incorporar nuevas tecnologías en el proceso de despacho.

7. Conclusiones

A lo largo de este documento se plantea una posible solución para facilitar la interacción de dispositivos de navegación Garmin nüvi con sistemas de gestión de flota. Luego de estudiar algunos casos de negocio de transporte, se llega a identificar y a modelar satisfactoriamente las entidades que participan directa o indirectamente en la problemática. Este análisis complementa la información técnica provista por la documentación del fabricante para comunicarse con el equipo, de tal forma que es posible construir un sistema que se ajuste a las funcionalidades básicas del proceso de despacho.

La implantación y puesta a prueba del sistema bajo condiciones reales de operación en el área de soporte de la compañía, permiten señalar que el objetivo general de esta tesis estaría cubierto de manera efectiva.

Por otro lado, los objetivos específicos de este trabajo, que tienen que ver con el aumento del nivel de satisfacción de los usuarios al utilizar el sistema, se comparan con los niveles reales de satisfacción, donde se evidencia un cumplimiento parcial:

- **Despachadores:** 20% esperado v/s 6% real.
- **Conductores:** 15% esperado v/s 10% real.
- **Administradores:** 10% esperado v/s 5% real.

Uno de los factores que puede incidir en cumplimiento parcial de los objetivos es el bajo nivel de madurez tecnológica y de los procesos de planificación y despacho que se realizan en el área evaluada. Las condiciones para llegar a obtener un cumplimiento mayor, tienen que ver principalmente con el mejoramiento del proceso de gestión de las visitas a terreno en el área de soporte. Los aspectos a considerar para este mejoramiento tienen que ver con:

- Generar una buena planificación de las visitas, con la anticipación suficiente.
- Formalizar las el proceso y actividades de cada rol.
- Hacer seguimiento del cumplimiento de las actividades.
- Canalizar solicitudes de visitas por vía única.
- Homogeneizar las herramientas de software utilizadas.

En el caso del área organizacional con la cual se trabajó, el rol del Administrador es el que tiene que facilitar y conducir los cambios necesarios en el proceso. En la encuesta todos los usuarios valoran cuán importante es la imagen de la empresa de cara al cliente, pero es a este usuario al que más le debería preocupar para impulsar mejoras efectivas.

Finalmente, queda para trabajo posterior estudiar si es posible o no mejorar el proceso de gestión de las visitas a terreno en el área de soporte. Podría ser el caso que ello no fuera posible con el actual personal y contexto. Actualmente se realiza investigación para apoyar procesos dinámicos y flexibles [33], que podría ser un camino a explorar.

Glosario

ACK: de la contracción en inglés “Acknowledgment”. Se refiere a un mensaje que informa que el grupo de datos ha llegado a su destino final sin problemas o errores.

AJAX: es el acrónimo de “Asynchronous Javascript and XML”, es decir: Javascript y XML Asíncrono. Es una técnica para el desarrollo de páginas web que implementan aplicaciones interactivas.

API: de las siglas en inglés “Application Programming Interface”, que significa “Interfaz de programación de aplicaciones”. Es el conjunto de funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

Árbol DOM: DOM o “Document Object Model” es un conjunto de utilidades específicamente diseñadas para manipular documentos XML y por extensión XHTML o HTML. DOM transforma el código XML en una estructura de nodos interconectados en forma de árbol para su manipulación más eficiente.

AVL: de las siglas en inglés “Automatic Vehicle Location”, que significa “Localización Vehicular Automatizada”. Se aplica a los sistemas de localización remota en tiempo real, basados generalmente en el uso de un GPS, GSM, Wifi y un sistema de transmisión que es frecuentemente un módem inalámbrico.

COF: Centro de Operación de Flota. Conjunto de operadores que supervisan el correcto funcionamiento de toda una flota de buses, en términos de frecuencia y recorrido, además de registrar e informar todos los parámetros de operación.

CRC: de la sigla en inglés ”Cyclical Redundancy Code”, que significa “Código de Redundancia Cíclica”. Se trata de un método matemático a través del cual, permite detectar errores en la información. Es comúnmente utilizado en la transmisión de datos.

Comandos AT: o también conocidos como “comandos Hayes”, es un lenguaje desarrollado por la compañía Hayes Communications que prácticamente se convirtió en estándar abierto de comandos para configurar y parametrizar módems.

ETA: de las siglas en inglés “Estimated Time of Arrival”, que se traduce como “Tiempo estimado de arribo”. El término se utiliza en el ámbito del transporte donde se necesite precisar un espacio de tiempo o la hora estimada de finalización de una operación.

Flota: en el contexto del transporte terrestre, se refiere a un grupo de vehículos que pertenecen a una misma unidad de gestión.

FMI: de las siglas en inglés “Fleet Management Interface”, que se traduce como “Interfaz de gestión de flota”. Es una API del fabricante Garmin para desarrollar aplicaciones que se comuniquen con sus dispositivos de navegación para vehículos.

Garmin: Empresa Suiza que desarrolla y fabrica dispositivos de GPS para el ámbito civil, principalmente para tránsito terrestre, aunque también naval y aéreo.

Geocercas: Se definen como una delimitación geográfica virtual. Al detectar una entrada o salida de un dispositivo en este espacio digital, el sistema puede desencadenar las acciones o eventos que se consideren pertinentes.

GIS: de las siglas en inglés “Geographic Information System” que significa “Sistema de Información Geográfico”. En el sentido amplio del término corresponde a cualquier aplicación capaz de capturar, almacenar, manipular, analizar, administrar y desplegar datos geoespaciales.

GPS: de las siglas en inglés “Global Positioning System”, que significa “Sistema de Posicionamiento Global”. Es un sistema de navegación por radio basado en satélites desarrollado y gestionado por el Departamento de Defensa de EE. UU. El GPS permite a los usuarios que viajen por tierra, mar y aire determinar su posición y velocidad, así como la hora, las 24 horas del día con cualquier situación meteorológica y en cualquier lugar del mundo. Las señales de GPS están disponibles para un número ilimitado de usuarios de forma simultánea. Cualquier persona puede usar los satélites GPS de manera gratuita.

GPRS: de las siglas en inglés “General Packet Radio Service”, que significa “Servicio General de Paquetes vía Radio”. Es una extensión del Sistema Global para Comunicaciones Móviles (Global System for Mobile Communications o GSM) para la transmisión de datos mediante conmutación de paquetes.

JSON: del acrónimo en inglés de “JavaScript Object Notation”, es un formato de texto ligero para el intercambio de datos. JSON [17] es un subconjunto de la notación literal de objetos de JavaScript aunque hoy, debido a su amplia adopción como alternativa a XML, se considera un formato de lenguaje independiente.

MDT: de las siglas en inglés “Mobile Data Terminal”, que significa “Terminal de Datos Móvil”. Es un dispositivo computarizado usado principalmente en vehículos de transporte público, taxis, vehículos militares o policiales, etc., para comunicarse con una oficina de despacho.

Modo PAD: término que proviene del acrónimo en inglés “Packet Assembler / Disassembler”. Se refiere al método de transporte de datos seriales asíncronos sobre redes de tipo TCP/IP.

NAK: de la contracción en inglés “Negative Acknowledgment”. Acuse de recibo electrónico que indica que los datos han llegado a su destino con errores.

OTA: de las siglas en inglés “Over The Air”. Corresponde al método de distribuir una nueva configuración a través de comunicación inalámbrica con los dispositivos.

POJO: del acrónimo en inglés “Plain Old Java Object”. Se refiere a una instancia de una clase que está libre de dependencias de frameworks particulares. Surge como una reacción en el mundo Java a los frameworks cada vez más complejos, y que requieren un complicado andamiaje que esconde el problema que realmente se está modelando.

REST: concepto que describe cualquier interfaz entre sistemas que utilice directamente HTTP para obtener datos o indicar la ejecución de operaciones sobre los datos, en cualquier formato (XML, JSON, etc.) sin las abstracciones adicionales de los protocolos basados en patrones de intercambio de mensajes.

RFID: proviene de las siglas en inglés “Radio Frequency Identification”. Es una tecnología de almacenamiento y recuperación de datos sin contacto cuyo propósito fundamental es transmitir la identidad de un objeto (similar a un número de serie único).

RS232: es una interfaz de datos serial muy utilizada para el intercambio de datos entre equipos computacionales y periféricos. Típicamente, la comunicación serial se utiliza para transmitir datos en formato ASCII. Para realizar la comunicación se utilizan 3 líneas de transmisión: (1) Tierra, (2) Transmitir y (3) Recibir.

SMS: siglas del término en inglés “Short Message Service”. Es un servicio disponible en los teléfonos móviles que permite el envío de mensajes cortos, conocidos como mensajes de texto.

Spring MVC: es un módulo para el framework de desarrollo Spring, que implementa el patrón de diseño Modelo-Vista-Controlador. Permite agilizar el proceso de construcción de aplicaciones web en Java gracias a los métodos nativos de generación y manipulación de solicitudes HTTP.

SOA: De las siglas en inglés “Service Oriented Architecture” que significa “Arquitectura Orientada a Servicios”. Es un paradigma de arquitectura para diseñar y desarrollar sistemas distribuidos. Permite la creación de sistemas de información altamente escalables que reflejan el negocio de la organización, a su vez brinda una forma bien definida de exposición e invocación de servicios (comúnmente pero no exclusivamente servicios web), lo cual facilita la interacción entre diferentes sistemas propios o de terceros.

Telemetría: sistema que permite la monitorización, medición y/o rastreo de magnitudes físicas o químicas a través de datos que son transferidos a una central de control. El sistema de telemetría se realiza normalmente mediante comunicación inalámbrica.

USB: de las siglas en inglés “Universal Serial Bus”. Es una tecnología que permite conectar periféricos a un equipo digital.

Bibliografía

[1] PIARC Committee on Intelligent Transport, ITS Handbook.
http://road-network-operations.piarc.org

[2] Asvin Goel, “Fleet Telematics Real-time management and planning of commercial vehicle operations”, 2008, ISBN 978-0-387-75105-4.

[3] E. D. Kaplan, “Understanding GPS: Principles and Applications”, Artech House Publishers, ISBN 0890067937, February 1996.

[4] R. J. Bates, GPRS: General Packet Radio Service, McGraw-Hill Professional, 1st Edition, ISBN 0071381880, November 12, 2001.

[5] Sindhuja Bharthepudi, “A Review of Low Cost Object Tracking System”, International Journal of Computer Science & Engineering Technology (IJCSET), November 2013 | Vol 3, Issue 11, 423-426.
http://ijcset.net/docs/Volumes/volume3issue11/ijcset2013031108.pdf

[6] K. Sandeep and Raja Vikram, "Monitored Fleet Management System using GPS", International Journal of Advanced Trends in Computer Science and Engineering (IJATCSE), Vol.2 , No.6, Pages : 224-231 (2013).
http://warse.org/pdfs/2013/spicetem201346.pdf

[7] O. O. Alharaki and F. S. Alaieri and A. M. Zeki, “The Integration of GPS Navigator Device with Vehicles Tracking System for Rental Cars Firms“, International Journal of Computer Science and Information Security (IJCSIS), Vol. 8, No. 6, September 2010.
http://irep.iium.edu.my/4208/

[8] “Garmin Fleet-ready Navigators“.
http://www8.garmin.com/solutions/mobile-resource-management/

[9] “Global GPS Market”, Marketsand Markets.
http://www.la-razon.com/suplementos/financiero/mercado-GPS-crecera_0_1784221698.html

[10] “Fleet Management Interface Overview”.
http://developer.garmin.com/fleet-management/overview/

[11] “SkyPatrol TT8750 Users Guide”, Revision 1.00 , Aug 2008.

[12] “Packet Assembler/Disassembler (PAD) Configuration and Use”, Revision 1.05, Feb 2007.

[13] FMI Protocol Support, “Features Supported by Each Protocol”.
http://developer.garmin.com/fleet-management/protocol-support/

[14] “Fleet-ready Navigators”.
http://www8.garmin.com/solutions/mobile-resource-management/supported-devices/

[15] “Fleet Management Developer Kit”.
http://developer.garmin.com/fleet-management/download/#download

[16] “Fleet Management Controller User Guide”, May 1, 2014.

[17] “Garmin Fleet Management Interface Control Specification”, June 10, 2014.

[18] Roy Thomas Fielding, “Representational State Transfer: REST”, Chapter 5.
http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm

[19] “Java JDK 1.8”.
http://www.oracle.com/technetwork/java/javase/overview/index.html

[20] “Spring Framework”.
https://spring.io/

[21] “JSON”.
https://www.json.org

[22] POJO, “Plain Old Java Object”.
https://www.martinfowler.com/bliki/POJO.html

[23] “MySQL 5.6”.
https://dev.mysql.com/doc/refman/5.6/en/

[24] “Apache Tomcat 8.0”.
https://tomcat.apache.org/tomcat-8.0-doc/index.html

[25] “Hibernate”.
http://hibernate.org

[26] “HIbernate Spatial”.
http://www.hibernatespatial.org

[27] “Release Notes Ubuntu 16.04”.
https://wiki.ubuntu.com/XenialXerus/ReleaseNotes

[28] “API Google Maps”.
https://developers.google.com/maps/

[29] “PHP”.
http://php.net

[30] “Javascript”.
https://en.wikipedia.org/wiki/JavaScript

[31] “Apache HTTP”.
https://httpd.apache.org

[32] “Codeigniter”.
https://codeigniter.com

[33] “Bootstrap Framework”.
https://getbootstrap.com

[34] “JQuery”.
https://jquery.com

[35] “MAVEN”.
https://maven.apache.org

[36] “Postman”.
https://www.getpostman.com

[37] Pedro Antunes, Nelson Baloian, Gustavo Zurita, José A. Pino: "Supporting People-driven, Dynamic and Geo-located Work Processes". S-BPM ONE 2018 Conference. Linz, Austria, April 2018.

Anexo A: Configuración AVL

Como se presentó en la sección 2.2, la programación del equipo Skypatrol TT8750 se realiza a través de comandos “AT”. La configuración inicial se ejecuta a través de una sesión de consola vía puerto serial.

A continuación se presentan los comandos adicionales realizados a una programación típica (Ver “Guía de Usuario Skypatrol” [11]), para la habilitación del modo PAD.

Instrucción	**Descripción**
AT+IPR=9600	Establece el baudrate a 9600
AT$PADDST=”190.161.69.71”,7000	Establece el destino de los paquetes PAD
AT$PADTO=10	Establece el valor de timeout
AT$PADCMD=0	Setea opciones de configuración para modo PAD
AT$ACTIVE=1	Establece el modo activo de la conexión PAD TCP
AT&W	Grabar configuraciones establecidas hasta ahora
AT$HOSTIF=2	Activa la sesión PAD TCP
AT$RESET	Reiniciar el equipo

Tabla A.1: Comandos AT del AVL para habilitar modo PAD.
(Tabla de autoría propia).

Una explicación más en detalle del modo PAD se puede obtener del documento “Packet Assembler/Disassembler (PAD) Configuration and Use” [12].

Anexo B: Mensajes Garmin FMI

En esta sección se presentan los principales comandos del protocolo FMI utilizados para este trabajo. Mayor nivel de detalle se encuentra en el documento “Garmin Fleet Management Interface Control Specification” [17].

B.1. Activación interfaz FMI

```
/////// ENABLE_FLEET_MANAGEMENT_PROTOCOL_REQUEST [0x0000]
//
//     10  a1  0e [00  00][05][00][01  80  02  80  0a  80  0b  80  0c  80] a8  10  03
//
//         00/80 01    Unicode support
//         00/80 02    A607 Support
//         00/80 0a    Driver passwords
//         00/80 0b    Multiple drivers
//         00/80 0c    AOBRD
//
//     0 DLE
//     1 Packet ID 161 (decimal)
//     2 Size of Enable Fleet Request
//     3-4 Fleet Management Packet ID 0 (decimal)
//     5 feature_count
//     6 reserved 0 (decimal)
//     7-8 Driver passwords (Enabled) 0x800A (hexadecimal)
//     9-10 Multiple drivers (Disabled) 0x000B (hexadecimal)
//     11 Checksum 2's complement of the sum of all bytes from byte 1 to byte 10
//     12 DLE 16 (decimal)
//     13 ETX 3 (decimal)

//    typedef struct /* D607 */
//    {
//        uint8 feature_count;
//        uint8 reserved; /* Set to 0 */
//        uint16 features[];
//    } fleet_features_data_type;
//
```

Figura B.1: Especificación de mensaje activación FMI.
(Imagen basada en referencia [17]).

Por defecto, el equipo Garmin que soporte el protocolo FMI no interpretará ningún comando hasta que se envía la instrucción señalada en la figura B.1.

B.2. Mensajería texto

```
//////// A602_SERVER_TO_CLIENT_OPEN_TEXT_MSG [0x0021]
//
//   10  21  18 [21  00][37  29  33  33][65  73  74  65  20  65  73
//   20  75  6e  6f  20  6e  75  65  76  6f  00]  78  10  03
//
//    typedef struct /* D602 */ {
//        time_type origination_time;
//        uchar_t8 text_message[];
//    } A602_server_to_client_open_text_msg_data_type;
//
```

Figura B.2: Estructura mensajería texto.
(Imagen basada en referencia [17]).

```
/////////// A602_TEXT_MESSAGE_ACK 0x0020
//
//    10  a1  1e [20  00][4f  84  32  2a][00][00  00  00][00  00  00
//    00  00  00  00  00  00  00  00  00  00  00  00  00][02  00  00
//    00] f0  10  03
//
//    typedef struct /* D602 */
//    {
//        time_type origination_time;
//        uint8 id_size;
//        uint8 reserved[3]; /* set to 0 */
//        uint8 id[/* 16 bytes */];
//        uint32 msg_ack_type
//    } text_msg_ack_data_type;
//
```

Figura B.3: Acuse de recibo de mensajería de texto.
(Imagen basada en referencia [17]).

B.3. Envío de puntos a visitar

```
/////// A603_SERVER_TO_CLIENT_STOP_MSG [0x0101]
//
//     10  a1  2e [01  01][55  16  5c  2c][55  e0  3b  e8  89  b8  ca
//     cd][01  00  00  00][50  43  20  46  61  63  74  6f  72  79  20
//     28  4c  6f  73  20  4c  65  6f  6e  65  73  20  31  33  36  29
//     00] a1  10  03
//
//    typedef struct { /* D603 */
//        time_type origination_time;
//        sc_position_type stop_position;
//        uint32 unique_id;
//        uchar_t8 text[/* variable length, null-terminated string */];
//    } A603_stop_data_type;
//
```

Figura B.4: Estructura mensaje de detención.
(Imagen basada en referencia [17]).

```
////////////// A603_STOP_STATUS_DATA [0x0211]
//
//    10  a1  0a [11  02][01  00  00  00][68  00][ff  ff] db  10  03
//
//    typedef struct  { /* D603 */
//        uint32 unique_id;
//        uint16 stop_status;
//        uint16 stop_index_in_list;
//    } stop_status_data_type;
//
```

Figura B.5: Mensaje de estatus de detención.
(Imagen basada en referencia [17]).

```
/////// A603_STOP_STATUS_RECEIPT [0x0212]
//
//    typedef struct {    /* D603 */
//        uint32 unique_id;
//    } stop_status_receipt_data_type;
//
```

Figura B.6: Acuse de recibo de mensaje de detención.
(Imagen basada en referencia [17]).

Anexo C: Resumen Requisitos

Esta sección se señalan los requisitos funcionales y no funcionales que dan forma al proyecto de software construido en esta tesis.

Identificador	RF01
Nombre	Autenticación de usuarios
Tipo	Funcional
Descripción	A través de nombre de usuario y clave se debe controlar el acceso de usuarios a la aplicación web del sistema.
Referencia	Sección 4.8

Identificador	RF02
Nombre	Gestión de vehículos, conductores, AVLs y equipos Garmin
Tipo	Funcional
Descripción	El sistema debe permitir administrar vehículos, conductores, AVLs y equipos Garmin.
Referencia	Sección 4.2, 4.8

Identificador	RF03
Nombre	Gestión entre entidades de negocio asociadas
Tipo	Funcional
Descripción	El sistema debe permitir administrar las asociaciones entre: vehículo y AVL, AVL y Garmin, Garmin y conductor.
Referencia	Sección 4.2, 4.8

Identificador	RF04
Nombre	Envío de mensajes de texto
Tipo	Funcional
Descripción	El sistema debe permitir el envío y recepción de mensajes de texto entre la aplicación web y el dispositivo Garmin.
Referencia	Sección 4.2, 4.3, 4.8

Identificador	RF05
Nombre	Gestión de detenciones
Tipo	Funcional
Descripción	El sistema debe permitir administrar las paradas para un determinado vehículo utilizando la georreferencia del destino a visitar.
Referencia	Sección 3.2, 4.2, 4.8

Identificador	RF06
Nombre	Integración con sistema de mapas
Tipo	Funcional
Descripción	El sistema debe utilizar un sistema de mapas para facilitar el ingreso de ubicaciones geográficas.
Referencia	Sección 4.8

Identificador	RF07
Nombre	Panel de estado de viajes
Tipo	Funcional
Descripción	El sistema debe proveer de un panel con el estado actualizado de los viajes (Completados, Pendientes, Cancelados, Inactivos).
Referencia	Sección 4.8

Identificador	RF08
Nombre	Integración con equipos Garmin
Tipo	Funcional
Descripción	El sistema debe interactuar con dispositivos de navegación Garmin.
Referencia	Sección 4.3

Identificador	RF09
Nombre	Traducción protocolo FMI
Tipo	Funcional
Descripción	El sistema debe traducir los mensajes del protocolo Garmin en instrucciones no propietarias.
Referencia	Sección 4.3

Identificador	RF10
Nombre	Construcción de servicio(s) para integración
Tipo	Funcional
Descripción	El sistema debe proveer de un mecanismo para proveer de interoperabilidad con plataforma de monitoreo de vehículos.
Referencia	Sección 3.3, 4.4

Identificador	RF11
Nombre	Utilización Modo PAD equipo AVL
Tipo	Funcional
Descripción	El sistema debe utilizar el equipo AVL para comunicarse con el dispositivo Garmin.
Referencia	Sección 4.3

Identificador	RF12
Nombre	Provisión automática de detenciones
Tipo	Funcional
Descripción	El sistema debe cargar de forma automática las detenciones en el equipo Garmin.
Referencia	Sección 4.2, 4.3

Identificador	RF13
Nombre	Actualización automática de estatus de detenciones
Tipo	Funcional
Descripción	El sistema debe actualizar de forma automática el estatus de cumplimiento de las detenciones.
Referencia	Sección 4.2, 4.3

Identificador	RNF01
Nombre	Operación eficiente ante concurrencia de dispositivos conectados
Tipo	No Funcional (Eficiencia)
Descripción	El sistema debe operar de manera eficiente, al menos con una cantidad de 4 dispositivos Garmin conectados de manera concurrente.
Referencia	Sección 3.4, 4.3, 4.9

Identificador	RNF02
Nombre	Bajos tiempos de respuesta para usuarios de Aplicación Web
Tipo	No Funcional (Desempeño)
Descripción	El sistema debe tener buenos tiempos de respuesta (< 3 s) al trabajar de forma simultánea con una cantidad mínima de 3 usuarios Web.
Referencia	Sección 3.4, 4.9

Identificador	RNF03
Nombre	Tolerante a intermitencia en redes GPRS
Tipo	No Funcional (Confiabilidad)
Descripción	Se debe construir un servidor TCP/IP que permita superar fallos de comunicaciones en redes moviles.
Referencia	Sección 3.4, 4.3

Identificador	RNF04
Nombre	Sincronización de hora del sistema
Tipo	No Funcional (Confiabilidad)
Descripción	El sistema debe utilizar servicios de sincronización de hora para ser precisos con la información del negocio.
Referencia	Sección 4.9

Identificador	RNF05
Nombre	Realización de respaldos
Tipo	No Funcional (Seguridad)
Descripción	Se debe considerar el respaldo de las configuraciones y base de datos al menos una vez al día.
Referencia	Sección 4.9

Identificador	RNF06
Nombre	Interacción con otros sistemas
Tipo	No Funcional (Interoperabilidad)
Descripción	El sistema debe ser capaz de exponer y consumir servicios de otras aplicaciones para compartir sus funcionalidades.
Referencia	Sección 3.3

Anexo D: Detalle encuesta

En esta sección se presentan las preguntas de la encuesta.

D.1. Preguntas Generales

1) ¿Cuál es su valoración general de la planificación de visitas a terreno?

a) muy baja b) baja c) regular d) buena e) muy buena

2) ¿Cuán conforme está usted con las estimaciones de tiempo derivadas de la planificación?

a) muy conforme b) conforme c) me es indiferente d) disconforme e) muy disconforme

3) ¿Cuán realistas son los tiempos estimados para una orden de trabajo?

a) muy poco b) poco c) regular d) mucho e) bastante

4) ¿Cuánta participación tiene ud durante la organización de visitas a terreno?

a) muy alta b) alta c) mediana d) casi nula e) nula

5) ¿La planificación se realiza con la anticipación adecuada?

a) nunca b) casi nunca c) a veces d) con frecuencia e) siempre

6) ¿Con qué frecuencia se generan modificaciones de última hora en la planificación de visitas a cliente?

a) siempre b) con frecuencia c) a veces d) casi nunca e) nunca

7) ¿Qué impacto le generan las modificaciones de última hora en la planificación de visitas a cliente?

a) muy bajo b) bajo c) nulo d) alto e) muy alto

8) ¿Cuán frecuente debe re-planificar en el momento para cubrir las órdenes pendientes de servicio?

a) siempre b) con frecuencia c) a veces d) casi nunca e) nunca

9) ¿Cuánto le favorece que se le comunique con anticipación los cambios de planificación?

a) nada b) casi nada c) a veces d) mucho e) bastante

10) ¿Cuánto le sirve que conozca con precisión la ubicación de los lugares de visita a clientes para la organización y ejecución del servicio?

a) bastante b) mucho c) algo d) casi nada e) nada

11) ¿Ha tenido alguna experiencia utilizando herramientas tecnológicas cuando se organizan y desarrollan visitas a terreno?
a) nunca b) casi nunca c) a veces d) con frecuencia e) siempre

12) ¿Cómo evaluaría las herramientas informáticas que la empresa dispone actualmente para facilitar el desplazamiento a destino?
a) muy mala b) mala c) normal d) buena e) muy buena

13) ¿Haría uso de aplicaciones informáticas más elaboradas para apoyarse durante la gestión y realización de visitas a terreno?
a) siempre b) con frecuencia c) a veces d) casi nunca e) nunca

14) ¿Cuánto le ayudaría en su trabajo si las visitas a terreno se gestionan íntegramente de forma digital?
a) nada b) casi nada c) poco d) mucho e) bastante

15) ¿Qué valoración le daría a una gestión de las órdenes de trabajo más expedita?
a) nula b) casi nula c) mediana d) alta e) muy alta

16) ¿Qué importancia le daría al cumplimiento y puntualidad de las visitas a terreno en una buena imagen de la empresa de cara al cliente?
a) muy alta b) alta c) baja d) muy baja e) nula

D.2. Preguntas a rol Despachador

17) ¿Cuánto tiempo invierte aproximadamente de su actividad regular a la planificación de visitas a cliente?
a) 0 a 10 mins b) 11 a 30 mins c) 31 a 60 mins d) 61 a 120 mins e) 121 mins y más

18) ¿Cuánto tiempo invierte de su actividad comunicando al conductor su destino?
a) 0 a 2 mins b) 3 a 5 mins c) 6 a 10 mins d) 11 a 30 mins e) 31 mins y más

D.3. Preguntas a rol Conductor

19) ¿Cuánto tiempo dedica (en promedio) a planificar la ruta de visitas a terreno?
a) 0 a 5 mins b) 5 a 10 mins c) 10 a 20 mins d) 20 a 30 mins e) 31 mins y más

20) ¿Cuánto mejoraría su planificación de tareas si tuviera una estimación de tiempos de desplazamiento?

a) nada b) casi nada c) poco d) mucho e) bastante

D.4. Preguntas a rol Administrador

21) ¿Cómo influiría en sus indicadores internos del área una mejor planificación de las órdenes de servicio en terreno?

a) muy alta b) alta c) baja d) muy baja e) nula

22) ¿Cómo influye en sus indicadores internos del área un menor tiempo de re-trabajo en la planificación de tareas?

a) nula b) muy baja c) baja d) alta e) muy alta

23) ¿Cómo evaluaría la gestión y cumplimiento actual de la planificación de visitas a cliente?

a) deficiente b) baja c) normal d) bueno e) destacable

24) ¿Cómo evaluaría la puntualidad en el cumplimiento de visitas a terreno?

a) destacable b) bueno c) normal d) baja e) deficiente

Anexo E: Análisis Servidor FMI

En esta sección se presentan segmentos del código fuente utilizado para implementar el Servidor FMI. Esta pieza de software es una aplicación en Java nativo que se ejecuta como cliente de la API Garmin y fue construido bajo la estructura de un proyecto maven (Ver figura E.1).

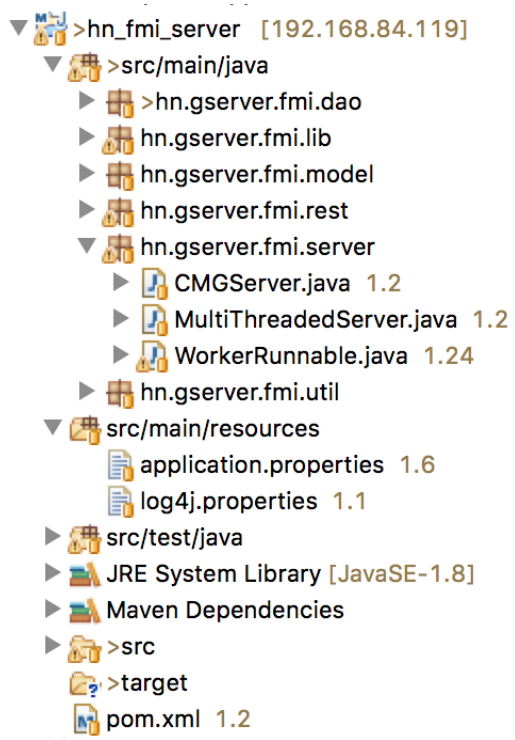

Figura E.1: Estructura de proyecto maven para la aplicación “Servidor FMI”. (Imagen de autoría propia).

Las variables parametrizables de este proyecto se encuentran en el archivo “application.properties” donde, se indica el puerto TCP a utilizar por el servidor y las configuraciones para comunicarse con la API Garmin. En la figura E.2 se muestra el detalle de los parámetros utilizados.

La clase principal del Servidor FMI se llama “CMGServer” y su tarea fundamental es preparar la aplicación para la ejecución bajo condiciones de multi concurrencia (Ver figura E.3). Las actividades a ejecutar son:

- Leer el archivo de configuración
- Inicializar el cliente REST
- Inicializar y ejecutar servidor multihilo

```
# Archivo de configuración
api.host = 127.0.0.1
api.port = 8090
api.name = hn_gserver_api
api.version = 1.4

server.port = 8123
server.source_id = 500
client.timeout = 90000

api.host.ext = 127.0.0.1
api.port.ext = 8090
api.name.ext = hn_gserver_api
api.version.ext = 1.4
```

Figura E.2: Parámetros de configuración del Servidor FMI.
(Imagen de autoría propia).

```
public class CMGServer {

    static RestManager restManager = null;
    static Configuration config;

    public static void main(String[] args)
        Throws ConfigurationException, FileNotFoundException {

        config = new PropertiesConfiguration("application.properties");
        boolean isRun = true;

        // Iniciando API Manager
        restManager = new RestManager();
        restManager.setConfiguration();
        logger.info("Starting REST Manager");

        // Iniciando Servidor multihilo
        MultiThreadedServer ser = new MultiThreadedServer(config.getInt("server.port"));

        ser.setRestManager(restManager);
        new Thread(server).start();
        logger.info("Starting Server");

        while ( isRun ) {

            try {
                Thread.sleep(15 * 1000);
            } catch (InterruptedException e) {
                logger.error(" ERROR: Excepción en clase principal.");
            }
        }

       logger.info("Stopping Server");
       ser.stop();
    }
}
```

Figura E.3: Clase principal CMGServer.
(Imagen de autoría propia).

```
public class WorkerRunnable implements Runnable {

    protected Socket clientSocket = null;
    private RestManager restManager = null;
    boolean runLoop = true;

    public void run() {

        RestClient restClient =  new RestClient(restManager.getUrlBase());
        restManager.setRestClient(restClient);

        try {
            InputStream input  = clientSocket.getInputStream();
            OutputStream output = clientSocket.getOutputStream();

            byte[] inputBuffer = new byte[512];
            List<byte[]> inputQueue = new ArrayList<byte[]>();
            List<Integer> inputQueueDelIndex = new ArrayList<Integer>();
            List<FmiMessage> outputQueue = new ArrayList<FmiMessage>();

            //  Para una nueva conexión, se realiza la iniciación de protocolo Garmin.
            //  - "Enable Fleet Management"
            //  - "Product Id And Support Data"
            //  - "Unit ID"
            initGarminProtocol( input, output, inputBuffer, inputQueue,
inputQueueDelIndex, outputQueue );

            while (runLoop) {

                // En la cola outputQueue están los paquetes hacia dispositivo Garmin.
                if ( outputQueue.size() > 0 ) {

                    for ( int i = 0; i < outputQueue.size(); i++ ) {
                        if ( outputQueue.get(i).raw != null )
                            output.write( outputQueue.get(i).raw );
                    }
                    outputQueue.clear();
                }

                // Lee las tramas de “InputStream” y las deja en en inputBuffer.
                if ( input.available() > 0  ) {

                    input.read(inputBuffer);
                    FmiParser.loadRawMessages(inputQueue, inputBuffer);
                    FmiParser.queueProcessing( restManager, inputQueue,
                        inputQueueDelIndex, outputQueue);
                }

              Thread.sleep(10 * 1000);
            }
            output.close();
            input.close();

        } catch (Exception e) { logger.error("Término de hilo."); }
    }
}
```

Figura E.4: Clase “MultiThreadedServer” encargada de interactuar con colas de entrada y salida.

(Imagen de autoría propia).

La clase “MultiThreadedServer” implementa un servidor multihilo, donde a su vez inicializa y ejecuta la clase encargada de atender las conexiones generadas desde el equipamiento embarcado. Dicha clase se llama “WorkerRunnable” y a través de la implementación de la interfaz “Runnable” de Java, se genera el método “run()” que contiene la lógica de esta clase (Ver figura E.4).

El bucle de la clase “WorkerRunnable” busca periódicamente mensajes nuevos en las colas de entrada (inputQueue) y salida (outputQueue). La llegada de mensajes nuevos del dispositivo Garmin está manejada con la invocación al método estático “queueProcessing” de la clase “FmiParser”.

Por ejemplo, para procesar un mensaje de consulta al estatus de una detención “A603_STOP_STATUS_DATA” (estructura de mensaje presentada en la figura B.5), se deben considerar siguientes atributos o campos del evento:

- **unique_id:** identificador único de 32 bits de la parada para cada dispositivo.
- **stop_status:** código del estatus del viaje asociado a la parada. Los valores pueden ser 100 (activo), 101 (terminado), 102, (inactivo no leído), 103 (inactivo leído) y 104 (eliminado).
- **stop_index_in_list:** representa la posición relativa de la parada en la lista interna del dispositivo Garmin. Puede utilizarse para gestionar la secuencia de los viajes realizados por el conductor.

El código del método “queueProcessing” asociado al procesamiento de este mensaje, se presenta en la figura E.5. De acuerdo a dicho algoritmo, primero que todo se debe identificar el tipo de mensaje a procesar. Esto se realiza capturando los valores de las variables “packet_id” y “fmi_packet_id”. En el caso del mensaje “A603_STOP_STATUS_DATA”, el valor de su identificador FMI es “0x0211” en hexadecimal.

A continuación se deben leer las siguientes variables propias para este tipo de mensaje. Luego, se realiza la comunicación con la API Garmin para actualizar el estatus de la parada. Si la parada se creó en el dispositivo Garmin, este método también genera el registro en el servidor.

El paso final para el procesamiento de este mensaje es generar el acuse de recibo hacia el equipo navegador. Esta tarea se realiza llamando al método “genA603StopStatusReceipt()”, el cual construye un mensaje del tipo “A603_STOP_STATUS_RECEIPT” (estructura del mensaje presentada en figura B.6). El objeto generado se deposita en la cola de salida para su posterior envío.

```
public static synchronized void queueProcessing ( RestManager restManager,  List<byte[]>
inputQueue, List<Integer> inputQueueDelIndex,  List<FmiMessage> outputQueue) {

    int packet_id;
    int fmi_packet_id;

    for(int i=0; i < inputQueue.size(); i++) {

        packet_id = FmiParser.getPacketID( inputQueue.get(i) );
        fmi_packet_id = FmiParser.getFMIPacketID( inputQueue.get(i) );

        ...

        /*  A603_STOP_STATUS_DATA [0x0211]  */
        else if (
                packet_id == FmiFactory.FLEET_MANAGEMENT_PACKET &&
                fmi_packet_id == FmiFactory.A603_STOP_STATUS_DATA  ) {

            Long unitId = device.getUnitId();
            int uniqueId = FmiParser.getInt32( inputQueue.get(i), 5 );
            int stopStatus = FmiParser.getUint16(inputQueue.get(i), 9);
            int stopIndexInList = FmiParser.getUint16(inputQueue.get(i), 11);

            StopData stopData = restManager.getStopDataByUnitIdAndUniqueId( unitId , uniqueId
);


            if ( stopData == null ) {
                stopData = new StopData();
                stopData.setUnitId( unitId );
                stopData.setUniqueId( uniqueId );
            }

            stopData.setStopStatus( stopStatus );
            stopData.setStopIndexInList( stopIndexInList );

            int status = restManager.updateStopStatus( unitId, stopData );

            inputQueueDelIndex.add(i);
            outputQueue.add( new FmiMessage("ACK",packet_id) );

            FmiMessage fm = new FmiMessage();
            fm.raw = FmiFactory.genA603StopStatusReceipt(uniqueId);
            outputQueue.add(fm);
        }
        ...
    }

    for ( int i = inputQueueDelIndex.size()-1 ; i >= 0; i-- )
        inputQueue.remove((int)inputQueueDelIndex.get(i));

    inputQueueDelIndex.clear();
}
```

Figura E.5: Extracto del método “queueProcessing” para procesamiento de mensaje “A603_STOP_STATUS_DATA”.

(Imagen de autoría propia).