Universidad Austral de Chile

Facultad de Ciencias de la Ingeniería

Escuela de Ingeniería Civil en Informática

PROTOTIPO DE SISTEMA DE SEGURIDAD Y ALERTA EN LÍNEA PARA VEHÍCULOS MOTORIZADOS UTILIZANDO DISPOSITIVOS MÓVILES CON ANDROID.

Proyecto para optar al título de
Ingeniero Civil en Informática

PROFESOR PATROCINANTE
JORGE ANTONIO MORALES VILUGRÓN
INGENIERO ELECTRÓNICO
MAGISTER EN ADMINISTRACIÓN, MBA

PROFESOR INFORMANTE
MARÍA ELIANA DE LA MAZA WERNER
INGENIERO CIVIL EN INFORMÁTICA
MAGISTER EN INFORMÁTICA EDUATIVA

PROFESOR INFORMANTE
LUIS ALVAREZ GONZALEZ
INGENIERO CIVIL ELECTRICISTA
MAGISTER EN INGENIERÍA INFORMÁTICA

JUAN GUILLERMO MOLT CANCINO

VALDIVIA – CHILE
2014

“Si buscas resultados distintos, no hagas siempre lo mismo.”

Albert Einstein

AGRADECIMIENTOS

A mi Familia
Sin su apoyo, esto no hubiera sido posible.

A la Mani
Por recordarme que termine la tesis, aunque ya lo sabía.

A Pamela
Por recordarme cada día que tenía que avanzar y no sacar la vuelta.

A Héctor
Por su ayuda con el video.

A Rodrigo
Por… bueno, ya que no me acuerdo por qué.

A Donald
Por los desayunos light en el McDonald's

A mi Patrocinante
Por todas las correcciones durante el semestre.

A los Informantes
Por su buena onda y disposición.

A mi Perro
Por recibirme moviendo la cola, sin importar la hora a la que llegue.

A mí Mismo
Porque a fin de cuentas, yo escribí este documento ☺

A los Innombrados
Los que me olvidé de nombrar antes, siéntanse agradecidos en esta línea.

ÍNDICE

ÍNDICE DE TABLAS

Tabla Página

ÍNDICE DE FIGURAS

Figura Página

RESUMEN

En la actualidad los servicios de monitoreo y ubicación geográfica de vehículos motorizados están orientados principalmente a la industria y al monitoreo de flotas. Trabajan mediante el uso de sofisticados dispositivos y comunicación bidireccional entre el objeto rastreado y los satélites de posicionamiento global.

Este proyecto de titulación, tiene como objetivo el diseño e implementación de un prototipo integrado software-hardware, que permita conocer el estado de un vehículo de forma remota. Es decir, que permita conocer el estado del vehículo cuando éste se encuentre estacionado en la vía pública, en un centro comercial, circulando en la carretera o en cualquier otro sitio donde pudiera estar el vehículo.

Esto permitirá saber si el vehículo continúa estacionado donde se dejó la última vez, si es que el sistema de seguridad del vehículo ha sido activado por algún evento externo, si se encuentra en movimiento y cualquier otra medición para lo cual se cuente con el sensor instalado al interior de éste.

Para el desarrollo del proyecto, se diseñará una unidad que se encargará de capturar datos del entorno, interpretarlos y transmitirlos a través de la red celular. Estos datos serán enviados por un dispositivo móvil con sistema operativo *Android*, que se encontrará instalado al interior del vehículo. Los datos transmitidos serán recibidos en el dispositivo móvil del propietario del vehículo, alertando oportunamente sobre la eventualidad que esté ocurriendo en el momento.

El objetivo es hacer uso de un teléfono inteligente y la red de telefonía móvil del país para crear un sistema económico y eficiente, para el monitoreo de automóviles particulares. Esta aplicación tiene como particularidad que podrá ser instalada en cualquier dispositivo con sistema operativo *Android*, pudiendo este último ser un teléfono o una *tablet*. Para llevar a cabo este proyecto, se debe diseñar y construir una unidad física de hardware para recibir e interpretar datos del ambiente. Desarrollar una aplicación para un dispositivo emisor, que interpretará los datos del ambiente y los transmitirá hacia un servidor central. Desarrollar e implementar los servicios que deberá atender éste servidor. Finalmente se debe desarrollar una aplicación cliente, la cual se instalará en el o los dispositivos móviles de los usuarios finales para recibir las alertas del sistema.

ABSTRACT

Nowadays the vehicle monitoring and vehicle geographic location services are principally oriented to the big industry and monitoring fleets. The system works with advanced devices and bidirectional communication between the tracked object and the GPS satellites.

This project for university degree, have as objective of work the design and deployment of a software-hardware integrated prototype to know the state of a vehicle remotely. In other words, know the state of the vehicle when it is left parked on the public site, in a store, driving by the highway or any other site.

The system will allow to the owner know if the vehicle is still in the parking site, if the security system has been activated by an external event, if the vehicle is moving or any other measure that counts with the necessary sensor installed.

For the project development, will be implemented a unit for capture data from the environment, process the data and send it through the mobile network. This data will be send by a mobile device with Android OS, installed inside the vehicle. The transmitted data will show and alert on the client device in real time.

The target is use a smartphone and the mobile network to implement an economic vehicle monitoring and tracking system. The app has the particularity that can be installed in any device provided with *Android OS*. It could be a smartphone or a tablet. To carry out this project, we must design and build a hardware device to receive and interpret environmental data. Develop an application which interpret environmental data and transmit them to a central server. Develop and implement services that will be serving in the server. And finally, we must develop a client application, which is installed on mobile devices or end users to receive the system alerts.

1. INTRODUCCIÓN

De acuerdo al último reporte *Parque de Vehículos en Circulación*, que se realiza cada 10 años, del Instituto Nacional de Estadísticas (INE) con cifras hasta el año 2010. Señala que durante ese año se registraron un total de 3.299.446 unidades motorizadas nuevas. La cifra representó un alza del 7,54% respecto de 2009 incluyendo transporte particular, colectivo y de carga. Si se compara el volumen del año 2010 con el de 2000, que llegó a 2,07 millones de vehículos con motor registrados ese año, el crecimiento fue de un 58,7% en 10 años.

Por otra parte, el gran desarrollo de los teléfonos móviles en la actualidad nos permite contar con teléfonos inteligentes o *smartphones* con gran capacidad de cómputo y procesamiento de información. Los *smartphones* prácticamente han desplazado del mercado a los teléfonos celulares convencionales. En Agosto de 2012, la venta de *smartphones* en Chile, superó por primera vez la venta de teléfonos convencionales [Ent12]. Además, junto con esto, entre los sistemas operativos para teléfonos inteligentes, el sistema operativo *Android* se convirtió en el más utilizado en los teléfonos inteligentes del mundo en el tercer trimestre de 2013, ampliando su ventaja sobre el iOS de Apple y el Windows Phone de Microsoft, de acuerdo con un estudio publicado por la consultora IDC la cuota de mercado de *Android* subió a 81% durante 2013. [Uni13]

De esta forma, de acuerdo a los datos recopilados, tenemos un parque automotriz en aumento y además sabemos que más de la mitad de las personas del país prefiere un teléfono inteligente por sobre un teléfono convencional. Y no solo eso, sino que sabemos que del total de personas que disponen de un teléfono inteligente, el 81% utiliza sistema operativo *Android*. La gran cantidad de vehículos en circulación, no solo llena las calles, sino que además aumenta las cifras de vehículos robados. La mayoría de los vehículos se encuentran equipados con sistemas de alarma activos o pasivos, pero solo alertan con sistemas sonoros, sirven más como un medio disuasivo, se necesita de personas en las cercanías del vehículo para alertar sobre el sonido emitido por la alarma de éste. En relación a los vehículos que son sustraídos en el país, de acuerdo a cifras entregadas por Carabineros de Chile, solamente durante el año 2012, fueron robados 32.742 vehículos en el territorio nacional. [Coo13]

La irrupción de nuevas tecnologías nos desafía a desarrollar sistemas que podrían ayudar en gran medida a la disminución de este tipo de delitos. Hoy existen diversos

tipos de sensores que permiten tomar datos provenientes del ambiente, por otro lado, el avanzado desarrollo de los teléfonos móviles en la actualidad, hace que cada persona lleve consigo una unidad computacional móvil en sus bolsillos. De ésta forma, resulta atractivo desarrollar un sistema de seguridad para vehículos motorizados, que incorpore éstas tecnologías existentes y las englobe en un mismo sistema para brindar un servicio adicional al usuario final.

1.1 Objetivos

Objetivo General

Diseñar y desarrollar un prototipo de un sistema de seguridad y alerta en línea para vehículos motorizados utilizando dispositivos móviles con *Android*^{TM}.

Objetivos Específicos

- Identificar situación actual y características de los sistemas de seguridad existentes para vehículos.
- Definir una arquitectura de software que se adapte al prototipo a desarrollar.
- Definir las tecnologías y requisitos para el desarrollo del prototipo.
- Implementar el prototipo del sistema de acuerdo a lo modelado en las etapas anteriores.
- Validar el sistema mediante pruebas de funcionalidad.

1.2 Motivación

- Escasa innovación en productos similares, muchas tecnologías distintas, pero no se ha encontrado un producto que incorpore las características de rastreo, monitoreo de estado y alerta en línea en un mismo sistema.

- El número de teléfonos inteligentes en usuarios finales es alto. Además la oferta es cada vez mayor, y es un dispositivo que ha pasado a ser un elemento utilizado a diario por usuarios finales.

- Los sistemas actuales de monitoreo por sistemas de posicionamiento global (GPS), están orientados al monitoreo de flotas y no al usuario común.

- La mayoría de los sistemas de seguridad en vehículos, son sistemas audibles preinstalados de fábrica, sólo alertan con alarmas sonoras.

1.3 Impactos

El impacto principal es el bajo costo del sistema, logrando que cualquier persona que ya disponga de un teléfono inteligente, pueda incorporar este sistema a su vehículo, sin tener que adquirir otros dispositivos de mayor valor ni pagar rentas mensuales por el uso de satélites. Solo requiere de un dispositivo móvil con plan de datos activo para poder utilizar este sistema.

Otra característica diferenciadora es que con el sistema propuesto, el usuario podrá saber que sucede con el vehículo en todo momento, a través de un software para dispositivos móviles, mediante el cual podrá recibir las alertas directamente en su teléfono.

El prototipo propuesto, tendrá la particularidad de poder integrarlo al sistema de seguridad presente en el vehículo, ya que los vehículos provistos con un sistema de alarma cuentan con sensores de apertura de puertas y quiebre de ventanas. De este modo, facilitaría la instalación.

2. ESTADO ACTUAL

2.1 Tipos de Alarmas para Vehículos

La protección de un vehículo frente a robo o hurto es responsabilidad del propietario del mismo. Actualmente existen diversos tipos de sistemas de seguridad con diferentes características y precios. Entre los diferentes tipos de alarmas de vehículos que pueden mejorar la seguridad del vehículo y la seguridad del propietario se distinguen principalmente las alarmas de fábrica y las alarmas del mercado. [Meg12]

Alarmas de fábrica: estos tipos de alarmas son ofrecidos por fabricantes de equipos originales (OEM[1]). Estos sistemas se instalan en los vehículos al momento de la construcción del mismo y pueden ser de dos tipos: Sistema Activo o Sistema Pasivo. [Meg12]

Las alarmas con sistema activo, requieren del propietario / usuario del vehículo para su activación antes de alejarse el vehículo. Una alarma con sistema activo puede ser activada desde un transmisor en el llavero del vehículo o un mecanismo al interior del mismo. Una desventaja es que el conductor tiene que recordar encender o activar la alarma. [Meg12]

A diferencia del sistema activo, las alarmas con sistema pasivo son mucho más inteligentes. Sólo con apagar el motor y cerrar las puertas del vehículo para ésta se enciende automáticamente. Por lo que no es necesario que el propietario / usuario del vehículo recuerde activarla al alejarse de él. Una ventaja de este tipo de sistemas, es que las compañías aseguradoras, ofrecen un descuento en la prima de seguros para vehículos con este tipo de sistemas. [Meg12]

Las alarmas de fábrica están disponibles en la mayoría de los vehículos, pero la desventaja de este tipo de alarmas es que un delincuente experimentado sabe exactamente cómo desactivar una alarma tradicional.

Las alarmas del mercado, son las que se instalan después de adquirir el vehículo. Estas alarmas incluyen todas las características de las alarmas de fábrica y agregan otras

[1] Un fabricante de equipamiento original (en inglés *original equipment manufacturer*, OEM), es una empresa que fabrica productos que luego son comprados por otra empresa y vendidos bajo la marca de la empresa compradora

características que hacen que el vehículo sea más seguro. Potenciar a un vehículo con alarmas, además de las de fábrica hace que el robo de éste sea más difícil, y aumenta el tiempo de los ladrones para intentar desactivar el armado del vehículo. Algunos de los sistemas más nuevos, incluyen funciones remotas, apagado del motor y localización por GPS [Meg12]. El rastreo por satélites de GPS se utiliza cada día más y varias empresas han comenzado a emplear estos sistemas para el rastreo de automóviles, autobuses y vehículos de trabajo.

2.2 Contexto Nacional e Internacional

En el contexto nacional, se ha identificado la empresa GPSTec[2] , que según describe su sitio web, ofrece un conjunto de servicios para el monitoreo de flotas empresariales. También se ha encontrado el sitio controlnet.cl[3] que ofrecen un servicio en tiempo real para el monitoreo de flotas y activos de grandes organizaciones.

A nivel internacional, el monitoreo de vehículos mediante sistemas de posicionamiento global o GPS está principalmente orientado al monitoreo de flotas. La metodología utilizada es la comunicación bidireccional de datos con satélites de baja órbita, donde un gran número, pertenecen a la empresa ORBCOMM[4]. Los vehículos deben poseer un dispositivo certificado y pagar por el uso de satélites. Los datos obtenidos son reservados. El software es propietario y está orientado fundamentalmente al seguimiento de flotas empresariales. [Upv13]

[2] Información disponible en www.gpstec.cl
[3] Disponible en www.controlnet.cl
[4] Empresa que provee comunicación a través de su red satelital. Más información en http://www.orbcomm.com/.

3. ARQUITECTURA Y TECNOLOGÍAS

3.1 Arquitectura del sistema

Debido a las características del sistema a desarrollar, se hace necesaria una arquitectura que integre las características del C*loud Computing* o cómputo en la nube para poder integrar a todos los dispositivos móviles de la solución como uno solo. Una representación abstracta del cómputo en la nube se muestra en la Figura 1.

Figura 1. Cloud Computing[5]

Cuando hablamos de tecnologías móviles, difícilmente podemos disociar el C*loud Computing* de la Arquitectura Orientada a Servicios conocida por sus siglas en inglés como SOA (*Service Oriented Architecture*). Ya que muchos de los servicios ejecutados en un dispositivo móvil son provistos por terceros como un SaaS (*Sofware as a Service*), en español, "Software como un Servicio". Los mapas, las notificaciones y la sincronización del reloj en un dispositivo móvil son ejemplos de SaaS.

La combinación de tecnologías móviles que se desarrollan bajo el paradigma SOA y el cómputo en la nube resulta en algo que hoy en día se conoce en inglés como *Mobile*

[5] Imagen obtenida de www.hightech-highway.com

Cloud Computing. En la Figura 2 se muestra un ejemplo de arquitectura MCC (*Mobile Cloud Computing*).

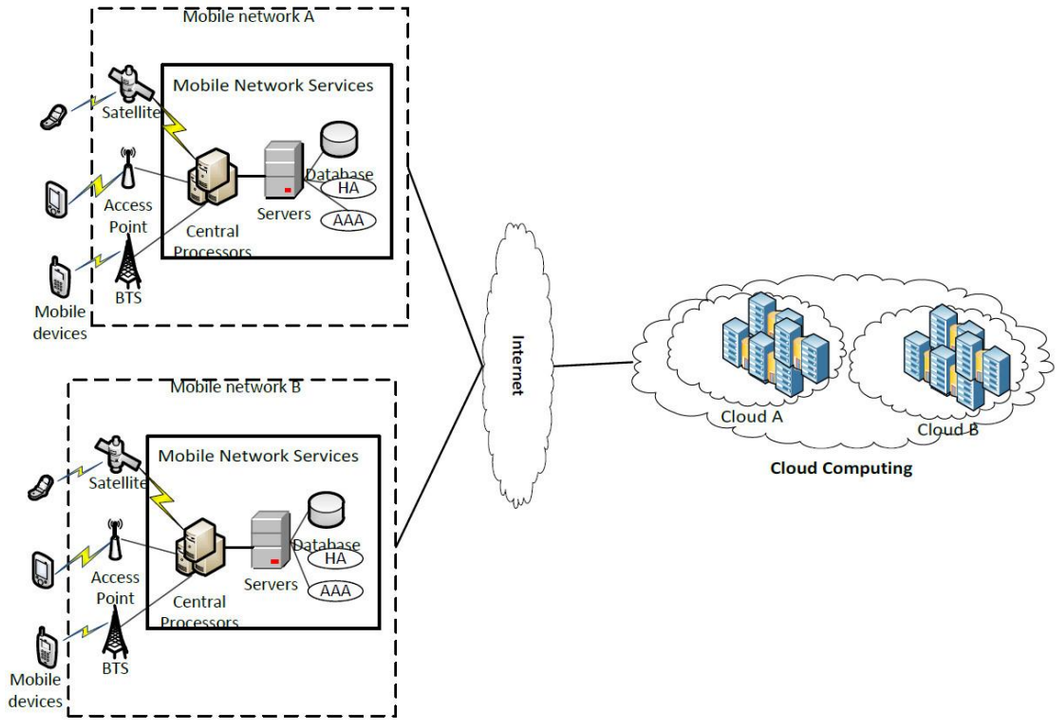

Figura 2. Arquitectura *Mobile Cloud Computing* (MCC).[6]

Para lograr el objetivo planteado se utiliza un teléfono con sistema operativo *Android* configurado como emisor que se ubica al interior del vehículo junto a una placa IOIO[7], conectada a algún(os) sensor(es) la cual sirve de nexo entre los sensores y el dispositivo móvil. Un servidor central ejecutando Apache, donde se recibe y procesa la información proveniente del emisor. Además de proveer y administrar los servicios de tipo SaaS provistos por Google. Y otro dispositivo móvil con sistema operativo *Android* configurado como receptor. Éste último es el dispositivo móvil del usuario final provisto con la aplicación desarrollada para éste sistema.

Las tecnologías utilizadas en la solución propuesta se muestran en la Figura 3, donde tenemos por un lado, los dispositivos que se encontrarán al interior del vehículo, algún(os) sensor(es), una placa IOIO y un dispositivo móvil. Lo anterior conectado a la fuente de energía del vehículo para proveer de energía permanente al sistema. En el centro del sistema, tenemos un servidor web central encargado de recibir y almacenar la información proveniente del emisor, además de enviar alertas al receptor. Y por otro

[6] Imagen obtenida desde http://en.wikipedia.org/wiki/Mobile_cloud_computing
[7] Tarjeta diseñada para trabajar con dispositivos *Android*, cuyo detalle se describe en el punto 3.2.2

lado, tenemos el dispositivo móvil configurado como receptor, donde se recibirán las alertas e información en línea del vehículo. Todos estos actores estarán comunicados mediante la red celular. Además de lo anterior, el sistema un portal web con el registro histórico de activaciones y alertas del sistema, además de la ubicación geográfica del vehículo.

Figura 3. Tecnologías asociadas a la solución propuesta.

Una característica de la solución propuesta, es que la placa IOIO tiene la capacidad de integrar diversos sensores dependiendo las necesidades del proyecto, en este proyecto son diversos los sensores que se podría incluir, entre los cuales, tenemos:

- Sensor de quiebre de cristales
- Sensor de apertura de puertas
- Sensor de presión en los asientos.
- Sensor de encendido/apagado del motor
- Sensor de movimiento

Para el desarrollo de este proyecto se implementan dos de los sensores mencionados anteriormente: el sensor de apertura de puertas y el sensor de presión en los asientos. Se determina incluir el de apertura de puertas, ya que de esta forma se tiene un control de acceso perimetral al vehículo. Otro sensor incluido en el prototipo, es el sensor de presión en los asientos, de esta forma, se puede determinar si han ingresado personas al interior del vehículo, incluso sin que se abran las puertas, es el caso en que se quiebra el

vidrio y se sienta para conducir el auto. Además el prototipo cuenta con un dispositivo que obtiene la ubicación del vehículo (latitud/longitud) y la informa oportunamente al servidor.

3.2 Tecnologías asociadas

3.2.1 Android

Android es un sistema operativo para dispositivos móviles basado en Unix, el cual está enfocado para ser utilizado en teléfonos inteligentes, televisores inteligentes, tabletas, *Android* TV y otros dispositivos. Está siendo desarrollado por la Open Handset Alliance, la cual es liderada por Google Inc.

Fue desarrollado inicialmente por Android Inc., pero esta fue absorbida por Google en 2005. *Android* cuenta con una larga lista de fabricantes que incorporan el SO en sus *smartphones*. De los fabricantes que incluyen el sistema operativo en sus teléfonos, Samsung es el que más ha contribuido al éxito de *Android*, con el 45,4 por ciento de todas las ventas de teléfonos inteligentes con éste sistema operativo. [Lib13]

La Tabla 1, muestra la participación de mercado de *Android* durante el año 2012.

Tabla 1. Participación del mercado de *Android* en EEUU[8]

Sistema Operativo	Unidades Vendidas 2012 (Millones de Unidades)	Porcentaje Mercado 2012
Android	497,1	68,8%
iOS	135,9	18,8%
Blackberry	32,5	4,5%
Symbian	23,9	3,3%
Windows Phone	17,9	2,5%
Otros	15,1	2,1%
Total	722,4	100%

Otra característica de *Android* es que tiene una gran comunidad de desarrolladores escribiendo aplicaciones para extender la funcionalidad de los dispositivos. A la fecha, se

[8] http://www.xatakamovil.com/mercado/android-y-ios-se-quedan-con-el-87-6-de-la-cuota-de-mercado-en-2012-segun-idc

han sobrepasado las 600.000 aplicaciones (de las cuales, dos tercios son gratuitas) disponibles para la tienda de aplicaciones oficial de *Android*, Google Play. [Red13]

En septiembre de 2013 se superaron los mil millones de dispositivos *Android* en todo el mundo desde que se pusiera a la venta el primer dispositivo *Android* allá por octubre de 2008. La consultora Gartner estima que sólo durante este 2014 se van a vender más de 1.000 millones de dispositivos *Android*. En otras palabras, lo que *Android* tardó en vender en cinco años lo podría vender este año en tan solo unos 12 meses. Con esta previsión en 2014 se superarían los dos mil millones de dispositivos *Android* en todo el mundo. [Xat14]

Debido a la gran cantidad de dispositivos que utilizan *Android* en el mundo, resulta interesante, desarrollar el prototipo del sistema para ésta plataforma, ya que de esta forma se llegará a un mayor número de usuarios potenciales.

3.2.2 Placa IOIO

El IOIO (pronunciado como yo-yo) es una tarjeta especialmente diseñada para trabajar con dispositivos Android (versión OS 1.5 o mayor). La tarjeta provee una conectividad robusta a cualquier dispositivo Android vía conexión USB y es totalmente controlable desde dentro de las aplicaciones Android usando una API de Java. [Spa13]

El IOIO contiene un pequeño microcontrolador que actúa como USB Host e interpreta las peticiones de la aplicación Android. Por otro lado, el microcontrolador puede interactuar con otros dispositivos físicos conectados al mismo tales como sensores, actuadores, etc. Utilizando las señales y protocolos convencionales tales como entradas y salidas digitales, entradas analógicas, SPI, UART, entre otros. [Oli13]

Para la realización de este proyecto, se utilizará la nueva versión de la placa IOIO de *SparkFun Electronics*, bautizada como IOIO-OTG la cual ya se encuentra disponible. Entre las nuevas características, ahora tiene un puerto mini-usb para la conexión con el dispositivo, además la nueva versión incluye un *jack* de conexión JST de 2 pines para la alimentación y cuenta con una nueva versión del *firmware*, por lo que se han actualizado las librerías y han cambiado ciertas instrucciones respecto de la versión anterior. En la Figura 4 se aprecia una tarjeta IOIO-OTG, indicándose los principales pines y partes.

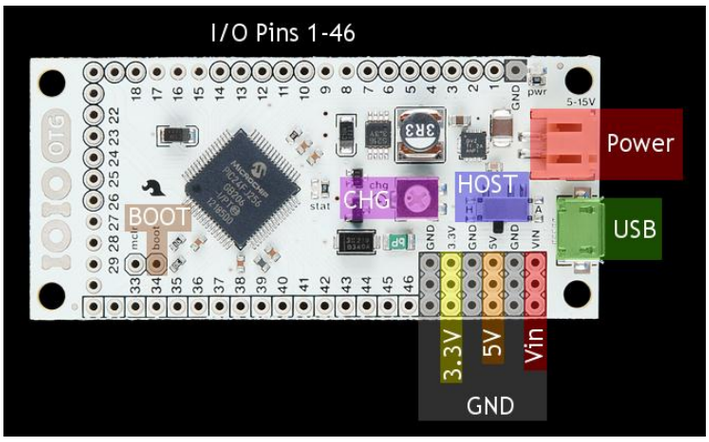

Figura 4. Placa IOIO-OTG y sus componentes[9].

3.2.3 Servidor Web Apache

En informática, un servidor es un tipo de software que realiza ciertas tareas en nombre de los usuarios. El término servidor ahora también se utiliza para referirse al ordenador físico en el cual funciona ese software, una máquina cuyo propósito es proveer datos de modo que otras máquinas puedan utilizar esos datos. [Per]

Un servidor sirve información a los ordenadores que se conecten a él. Cuando los usuarios se conectan a un servidor pueden acceder a programas, archivos y otra información del servidor. [Per]

El servidor HTTP Apache es un servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.12 y la noción de sitio virtual. [Net13]

Apache tiene amplia aceptación en la red: desde 1996, Apache, es el servidor HTTP más usado. Alcanzó su máxima cuota de mercado en 2005 siendo el servidor empleado en el 70% de los sitios web en el mundo, sin embargo ha sufrido un descenso en su cuota de mercado en los últimos años. [Net13]

Además de Apache, entre los servidores HTTP, el más conocido por el lado de Microsoft, es *Internet Information Services*, conocido por el acrónimo, IIS. Es un servidor HTTP de la familia de Microsoft Windows, actualmente es compatible con los sistemas operativos Windows Vista RTM, Windows Vista SP1, Windows XP, Windows

[9] https://github.com/ytai/ioio/wiki/Getting-To-Know-The-IOIO-OTG-Board

Server 2003 y Windows Server 2008. De acuerdo a Netcraft es el segundo servidor web más usado, después de Apache. [Net13]

Para la realización del presente proyecto, se determinó, que se utilizará el servidor HTTP Apache, bajo un entorno Unix. Debido primero, a que el servidor HTTP Apache es de código abierto, bajo licencia GNU GPL. Y segundo, que está basado en un sistema Unix/Linux, al igual que el sistema operativo Android, que también se utiliza para este proyecto.

3.2.4 PHP

Como lenguaje de programación para el servidor de este proyecto, se utilizará PHP, que es un lenguaje interpretado, de código abierto, y que está especialmente orientado al desarrollo web. Lo más importante, es que este lenguaje es correctamente soportado por el servidor web Apache, permitiendo un correcto funcionamiento al momento de implementar estas tecnologías de manera conjunta [Php].

3.2.5 MySQL

MySQL es un sistema de gestión de bases de datos de bases de datos relacional, multi-hilo y multiusuario con más de seis millones de instalaciones. Además de la facilidad de uso y el alto rendimiento. MySQL es muy utilizado en aplicaciones web, como Drupal o phpBB, en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), gracias a la disponibilidad de herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece en combinación con MySQL. [Ora14]

En sus inicios, Sun Microsystems y luego Oracle Corporation, desde abril de 2009 desarrolla MySQL como software libre en un esquema de licenciamiento dual. Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, desarrollos personales, código libre, etc. Pero en caso de utilizarla en sistemas privativos se debe comprar a la empresa una licencia específica que les permita este uso. [Ora14]

Entre los requisitos que cumple MySQL, es mantener el uso compatible con licencia GNU GPL, además de ser un sistema de gestión de bases de datos bastante seguro y robusto, compatible con entornos Unix/Linux.

3.2.6 Servicios Web SOAP

Existen múltiples definiciones sobre lo que son los Servicios Web, lo que muestra su complejidad a la hora de dar una adecuada definición que englobe todo lo que son e implican. Una posible sería hablar de ellos como un conjunto de aplicaciones o de tecnologías con capacidad para interoperar en la Web. Estas aplicaciones o tecnologías intercambian datos entre sí con el objetivo de ofrecer unos servicios. Los proveedores ofrecen sus servicios como procedimientos remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la Web. [W3C14]

Estos servicios proporcionan mecanismos de comunicación estándares entre diferentes aplicaciones, que interactúan entre sí para presentar información dinámica al usuario. Para proporcionar interoperabilidad y extensibilidad entre estas aplicaciones, y que al mismo tiempo sea posible su combinación para realizar operaciones complejas, es necesaria una arquitectura de referencia estándar. [W3C14]

En todo este proceso intervienen una serie de tecnologías que hacen posible esta circulación de información. Por un lado, estaría SOAP (Protocolo Simple de Acceso a Objetos). Se trata de un protocolo basado en XML, que permite la interacción entre varios dispositivos y que tiene la capacidad de transmitir información compleja. Los datos pueden ser transmitidos a través de HTTP, SMTP, etc. SOAP especifica el formato de los mensajes. [W3C14]

Por otro lado, WSDL (Lenguaje de Descripción de Servicios Web), permite que un servicio y un cliente establezcan un acuerdo en lo que se refiere a los detalles de transporte de mensajes y su contenido, a través de un documento procesable por dispositivos. WSDL representa una especie de contrato entre el proveedor y el que solicita. WSDL especifica la sintaxis y los mecanismos de intercambio de mensajes. [W3C14]

Además, en los servicios web SOAP, se destaca lo estricto del protocolo de comunicación. Ofrece una gran robustez y confianza en el sistema. Esta es la razón para utilizar servicios web SOAP en este proyecto. Ya que proporciona una mayor seguridad para el intercambio de mensajes entre los dispositivos remotos y el servidor central. [Gee09]

3.2.7 Mensajería Push

Cuando creamos aplicaciones que consumen datos desde Internet, a veces, nos encontramos con que necesitamos que nuestra aplicación tenga la capacidad de notificar eventos al usuario que ocurren fuera del dispositivo. Para lograr esto, existen varias opciones, una opción es mantener una conexión permanente con el servidor con el cual deseamos sincronizar, esto es viable y efectivo, pero consume demasiados recursos del dispositivo, principalmente tiempo de CPU y batería. Otra opción, es la encuesta, que trata de estar preguntando cada cierto tiempo al servidor si es que hay información nueva. Esto podría ser un gasto innecesario de red y CPU, ya que los datos podrían no haber cambiado aun, cuando se realiza la consulta. Esta última opción, también presenta otro inconveniente, ya que los datos podrían cambiar, y el dispositivo no tomaría conocimiento de éste cambio hasta la próxima consulta. Una opción que soluciona los problemas de uso innecesario de datos y batería, es un sistema de Mensajería Push. Esto significa, que el servidor es el encargado de avisar al dispositivo cuando ocurra un nuevo evento. Para los dispositivos Android, Google Play Services, proporciona un sistema de Mensajería Push para servidores, conocido como Cloud to Device Messaging (C2DM) o Mensajería C2DM.

Esta última opción, la Mensajería Push, cumple los requisitos de no consumir demasiado tiempo de CPU del dispositivo, y por consiguiente, no consumir demasiada batería. Para implementar un sistema de mensajería push, se debe utilizar la API de Google y desarrollar un sistema en un servidor, para administrar el servicio de mensajería. Un esquema más simplificado de un sistema de mensajería push tiene dos fases, la primera fase, que es el proceso de registro del dispositivo, se muestra en la Figura 5, y la segunda fase, es el envío del mensaje, como se muestra en la Figura 6.

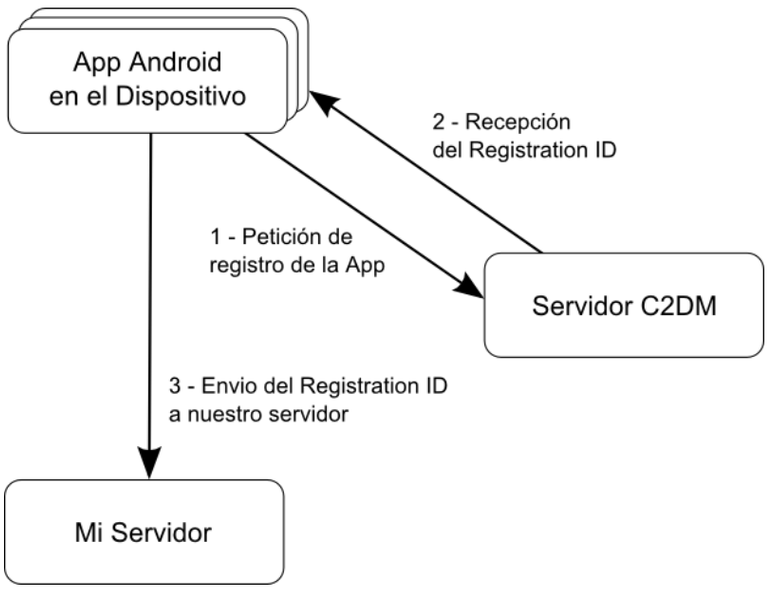

Figura 5. Registro de la Aplicación en el Servidor C2DM.

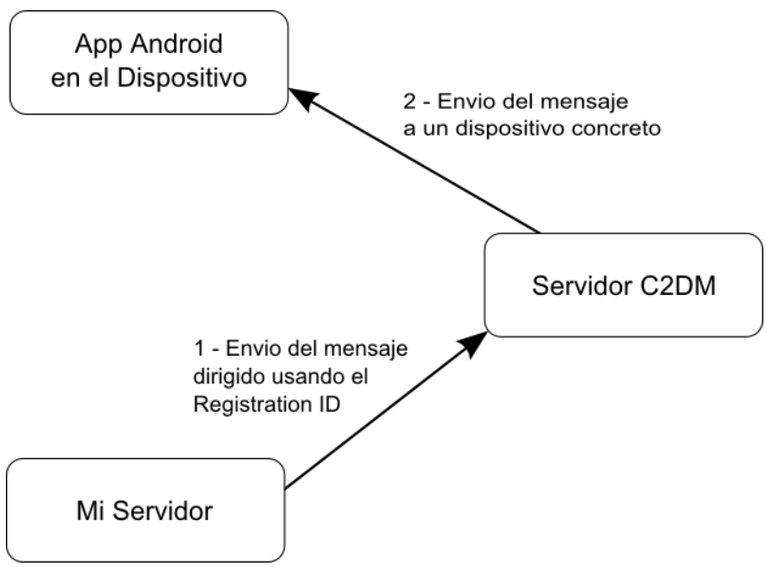

Figura 6. Envío de Mensaje Push hacia el dispositivo.

4. METODOLOGÍA

La elección del ciclo de vida de un proyecto permite administrar correctamente sus etapas con el fin de generar altas posibilidades de éxito. Además entrega herramientas para identificar las principales dificultades del proyecto, para así trabajarlas por separado, facilitando el entendimiento de la problemática y el posterior desarrollo del software.

En el marco de desarrollo de un proyecto de tipo software/hardware se pueden identificar claramente dos etapas durante su ejecución.

La primera etapa consiste en un prototipo del hardware que se utiliza para la solución, donde se realizan experimentos y pruebas con el hardware disponible para determinar el alcance de la tecnología. El ciclo de vida utilizado en la primera etapa de este proyecto es del tipo prototipado evolutivo. Este ciclo de vida consiste en el desarrollo de un prototipo inicial el cual es sometido a la evaluación del cliente para identificar las consideraciones que se harán para la siguiente versión del prototipo. Este proceso se realiza hasta que el prototipo cumple con todas las necesidades establecidas por el cliente. [Som09]

Mientras que la segunda etapa, consiste en el desarrollo de un software de control para el circuito implementado en la etapa anterior, el que puede estar compuesto de una interfaz para el operador, una interfaz para el usuario final, o ambas. El ciclo de vida o metodología utilizada para la segunda etapa de este proyecto es la metodología de desarrollo en cascada. Este ciclo de vida consiste en que las fases caen como cascada de una en otra, por ello su nombre. La segunda fase no debiera empezar hasta que la primera se termine, pero en la práctica las etapas se superponen e intercambian información con las otras [Som09]. Se determina usar esta metodología ya que el desarrollador tiene conocimientos previos del desarrollo de aplicaciones móviles y además se adapta al margen de tiempo determinado para el proyecto.

En la primera etapa, se realizan pruebas con distintas configuraciones de circuitos en placas de desarrollo, conocidas también como placas de *prototiping* o *protoboards*. La idea es lograr un desarrollo exploratorio, donde el objetivo del proceso es trabajar con el

cliente para explorar sus requerimientos y entregar un sistema final. El sistema evoluciona agregando nuevos atributos propuestos por el cliente. [Som09]

En la segunda etapa, se deben diseñar las componentes de software del proyecto. Se distinguen tres componentes software en el proyecto, el software del dispositivo emisor, el software del dispositivo receptor/cliente y el software del servidor. El software del emisor, es el encargado de obtener datos del ambiente a través de los sensores. El software del receptor o cliente es el encargado de proveer una interfaz de usuario con la información obtenida por el emisor. Mientras que el software del servidor es el que debe interactuar a través de servicios web y rutinas, con los otros dos dispositivos y la base de datos del mismo.

5. DESARROLLO

5.1 Requisitos Funcionales

En la Tabla 2, su muestran los requisitos funcionales del prototipo a desarrollar.

Tabla 2. Requisitos Funcionales del Sistema

Nº Ref.	Función	Tipo
R0	El sistema debe validar al usuario con nombre de usuario y contraseña cuando ejecute por primera vez.	Evidente
R1	El dispositivo emisor debe capturar datos de los sensores y enviar esta información al servidor central.	Evidente
R2	El dispositivo emisor, para determinados eventos de los sensores, debe enviar los datos al servidor central y levantar una alerta, para que los datos sean procesados como alerta.	Evidente
R3	El dispositivo receptor debe estar permanentemente preparado para recibir una alerta desde el servidor central.	Evidente
R4	El dispositivo receptor, debe ser capaz de recopilar y mostrar los datos almacenados en el servidor central.	Evidente
R5	El usuario, debe poder activar o desactivar el sistema en cualquier momento.	Evidente
R6	El usuario debe poder ver el listado de alertas del sistema, con fecha y hora.	Evidente
R7	El usuario debe poder visualizar la localización geográfica del dispositivo emisor en todo momento.	Evidente

5.2 Requisitos No Funcionales

En la Tabla 3, su muestran los requisitos no funcionales del prototipo a desarrollar.

Tabla 3. Requisitos No Funcionales del Sistema

Nº Ref.	Función	Tipo
R8	El software debe ser desarrollado para la plataforma *Android* en sus versiones 2.x y 4.x	Evidente
R9	El servidor central debe estar ejecutando un entorno Linux Ubuntu, con servidor web Apache y motor de bases de datos MySQL	Evidente

5.3 Pruebas de Concepto

A continuación se muestran los experimentos realizados con el prototipo para comprobar la factibilidad de la solución. De esta forma se prueban ciertas funcionalidades que son partes clave del sistema para que el desarrollo del prototipo pueda ser llevado a cabo en los términos preestablecidos.

Los elementos hardware y software que se utilizan para la mayor parte de los experimentos son los que se muestran a continuación. En algunos casos se agregaron otros dispositivos, los que están señalados en cada experimento.

Hardware Utilizado

- Placa IOIO-OTG
- Notebook Packard Bell Intel Core 2 Duo de 1.4 Ghz (para desarrollo)
- Servidor del Instituto de Informática (para montar servidor web y base de datos)
- Smartphone Motorola Razor D1 XT914 con Android 4.1.2 (emisor/servidor)
- Smartphone Samsung Galaxy Core I8260 con Android 4.1.2 (receptor/cliente)

Software Utilizado en Servidor del Instituto

- Sistema Operativo Ubuntu 12.04 Server
- Servidor HTTP Apache Versión 2.2.22

- Motor de Base de Datos MySQL Versión 5.5.37

Software Utilizado en PC de desarrollo

- Sistema Operativo Ubuntu 12.04 LTS
- IDE para desarrollo Eclipse v4.2 Juno
- Control de Versiones GIT, con respaldo web en github.com

5.3.1 Prueba de Concepto Nº 1 "Crear aplicación en Android que controle un elemento del ambiente mediante la placa IOIO"

Para realizar esta prueba, se desarrolla una pequeña aplicación con un botón táctil en la pantalla del teléfono, y se conecta a un LED a través de la placa IOIO. El objetivo es encender el LED al presionar el botón en la pantalla del teléfono.

El esquema de conexión se muestra en la Figura 7.

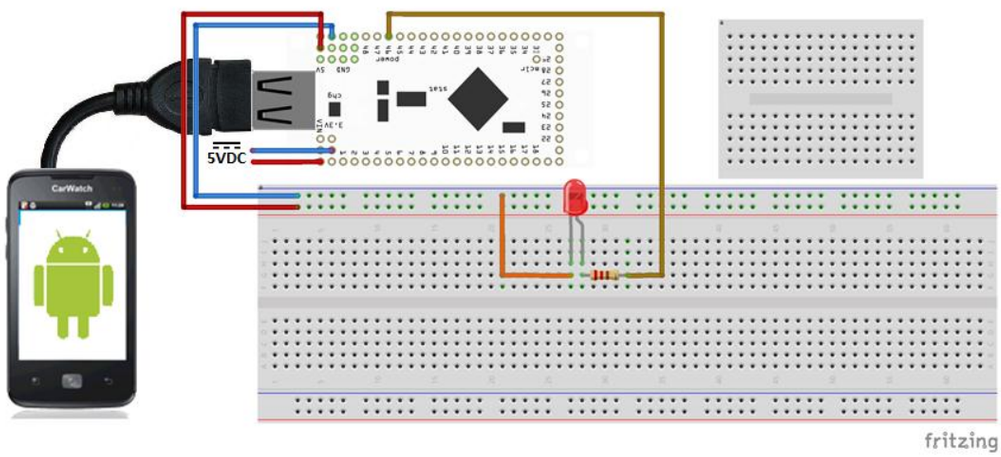

Figura 7. Esquema de Conexión Prueba de Concepto Nº1

Se utiliza como ayuda los códigos de ejemplo de la página Web Oficial de IOIO en Github[10] y se realizan modificaciones posteriores, en base a la API de la librería IOIO, ya que los códigos de ejemplo publicados en la web son para la versión antigua de la placa IOIO, y no funcionan con IOIO-OTG. Por ejemplo las clases *IOIOThread y IOIOActivity* fueron descartadas en esta nueva versión de la librería.

[10] https://github.com/ytai/ioio/wiki

Finalmente se logra el objetivo de controlar el encendido de un LED desde un botón táctil en el teléfono. En la Figura 8 se aprecia el teléfono ejecutando la aplicación y en la Figura 9, se puede ver que el objetivo del experimento se logra. Un fragmento de código de esta aplicación está disponible en el Anexo A de este documento.

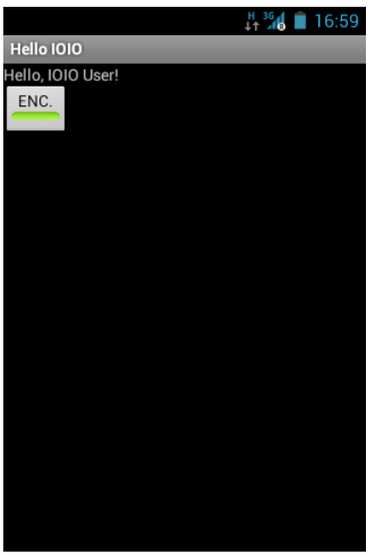

Figura 8. Aplicación para encender LED desde el teléfono.

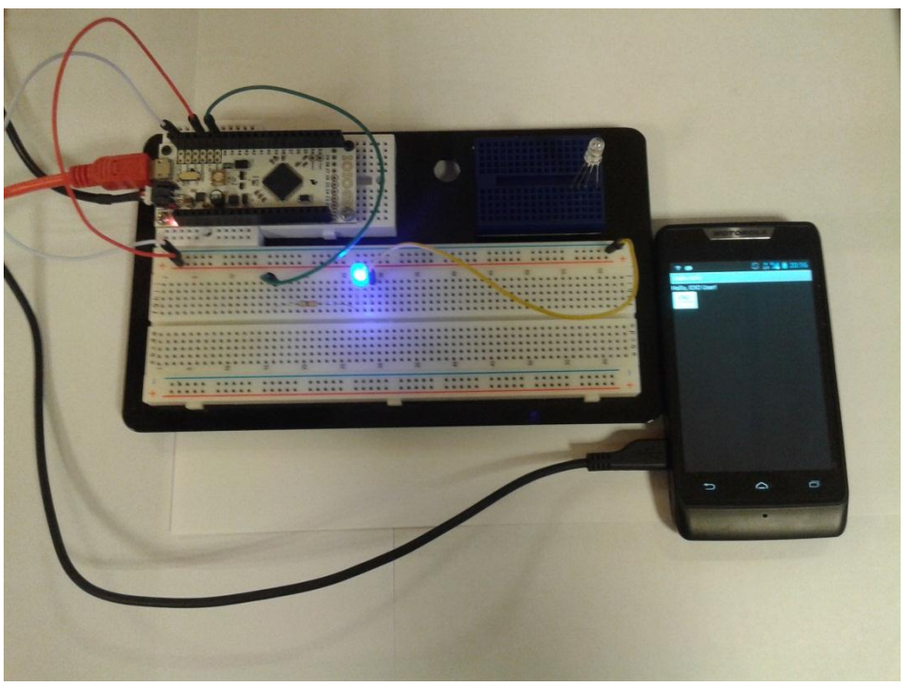

Figura 9. Aplicación y Esquema de encendido de LED.

5.3.2 Prueba de Concepto Nº 2 “Crear aplicación en Android que reciba un estímulo del ambiente a través de la placa IOIO”

Para la realización de este experimento se utiliza el esquema de conexión que se muestra en la Figura 10. Se conecta un *switch* magnético, a la placa IOIO y esta a su vez va conectada al dispositivo emisor. Se desarrolla una pequeña aplicación que sense cada cierto tiempo, el estado del *switch* magnético. El objetivo de este experimento es que el dispositivo emisor avise enviando un mensaje de texto a un dispositivo preestablecido cuando el *switch* sea activado.

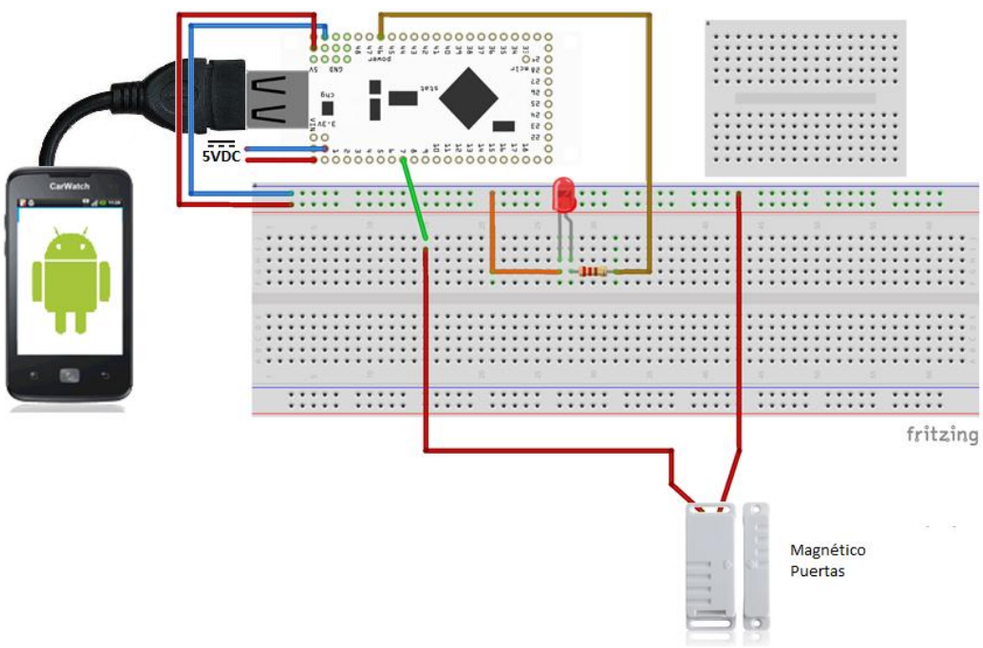

Figura 10. Esquema de Conexión Prueba de Concepto Nº2

En la Figura 11 se muestra en dispositivo emisor, ejecutando la aplicación desarrollada. Y en la Figura 12, se muestra el mensaje de texto recibido por el dispositivo cliente cuando el *switch* magnético es activado.

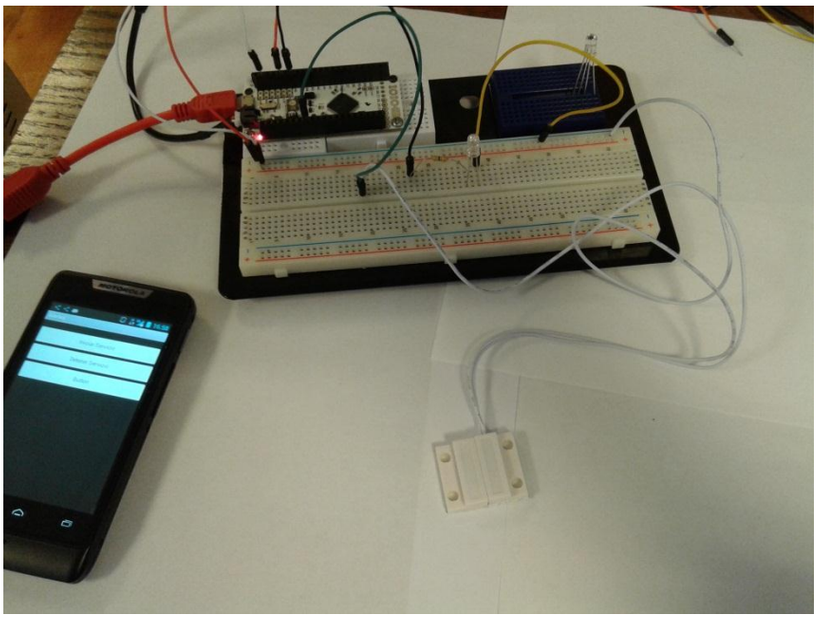

Figura 11. Aplicación y Esquema de *Switch* Magnético

Para facilitar la lectura, se agrega al experimento un LED que se enciende cuando los magnéticos son activados (son separados entre sí). De esta forma se puede saber si el circuito funciona correctamente. En la Figura 12 podemos ver que el objetivo del experimento se cumple.

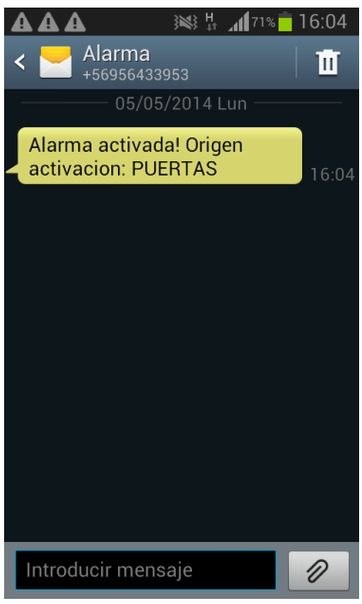

Figura 12. SMS recibido cuando se activa el *switch*

5.3.3 Prueba de Concepto Nº 3 “Enviar Mensaje Push cuando ocurra un evento en el dispositivo emisor”

Para realizar este experimento, se conecta un pulsador a través de la placa IOIO al dispositivo emisor. Además se agrega un LED indicador, el que tiene que parpadear cuando se accione el pulsador. El objetivo del experimento, es que el emisor envié un Mensaje Push al dispositivo receptor, haciendo uso de la API de Google y el servidor web del Instituto de Informática.

Como base para este experimento, se cuenta con los códigos de ejemplo de Google, en el ejemplo se utiliza un servidor web IIS (Internet Information Services) montado sobre un servidor con Windows y una base de datos SQL Server de Microsoft para crear un sistema de Mensajería Push.

Como primer desafío de este experimento, es hacer un *port* o *porting* del código de ejemplo, que es hacer una traducción y adaptación del lenguaje de programación, para que éste funcione en la arquitectura definida para el prototipo. De ésta forma las sentencias SQL que estaban diseñadas para SQL Server, tienen que ser portadas a MySQL. Y el código que se encuentra en .NET debe ser portado a PHP.

Luego de varios intentos y pruebas, se logra portar el código a la arquitectura del prototipo, uno de los problemas principales, es que no se tenía permisos suficientes en algunos directorios del servidor web. En el Anexo B.1 se muestra el código de ejemplo en lenguaje .NET y en el Anexo B.2 se muestra el código portado a PHP. En la Figura 13 se observa el esquema de conexión para esta prueba.

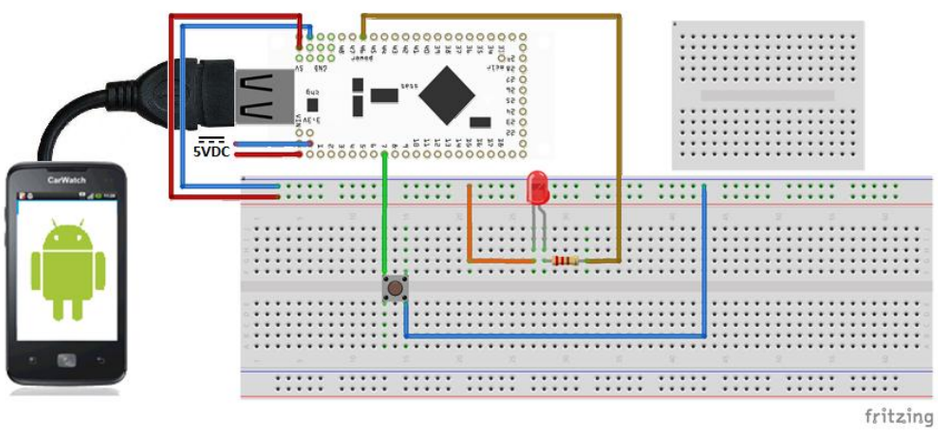

Figura 13. Esquema de Conexión Prueba de Concepto Nº3.

En la Figura 14 se muestra la aplicación funcionando y en la Figura 15 se puede ver el mensaje que llega al dispositivo cliente cuando se activa el pulsador en el emisor. Con esto podemos ver que se cumple el objetivo de este experimento.

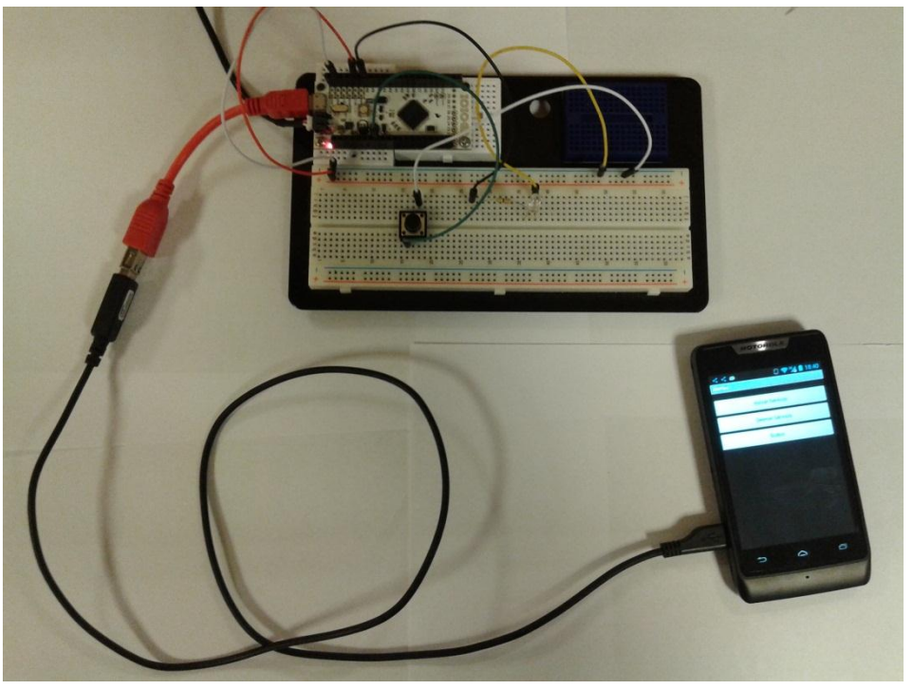

Figura 14. Aplicación de Mensajería Funcionando.

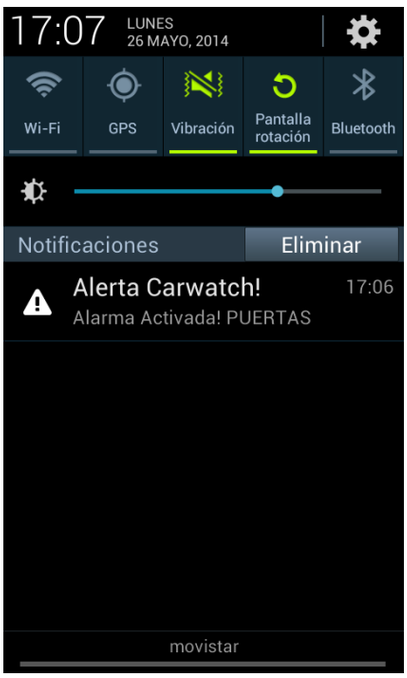

Figura 15. Mensaje Push recibido por dispositivo Cliente.

5.3.4 Prueba de Concepto Nº 4 “Prueba de Compatibilidad en Distintos Dispositivos Android”

El objetivo de este experimento es comprobar la compatibilidad de ciertas características de la aplicación cliente en diferentes dispositivos *Android*. En particular el sistema de *Tabs* (Pestañas) y el sistema de Menú Emergente. Ya que se utilizan distintas librerías para Android 2.x y en Android 4.x

Existen dos formas de crear *Tabs* en Android. Una forma es creando el objeto *Tab* en una Actividad y la otra es crear un TabFragment. Primero se prueba con un TabFragment, pero la aplicación no funciona en Android 2.x debido a que Android 2 no tiene soporte nativo para Fragments, por lo que se decide utilizar la otra forma mencionada con anterioridad, para tener una aplicación con mayor compatibilidad. La Figura 16 muestra el sistema de pestañas en Android 2.x y la Figura 17 muestra el sistema de pestañas en Android 4.x

Figura 16. Aplicación con Pestañas en Android 2.x

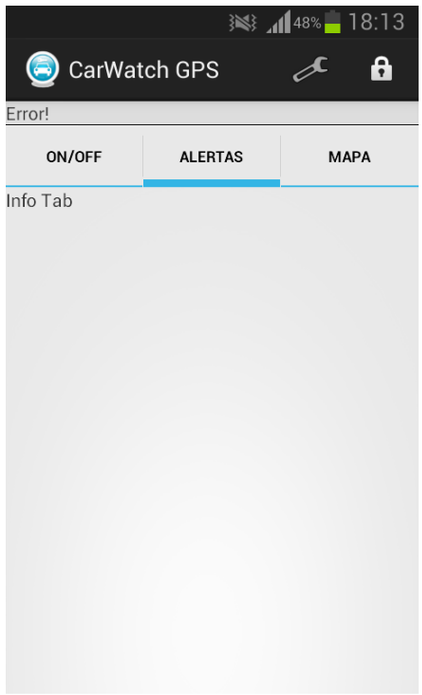

Figura 17. Aplicación con Pestañas en Android 4.x

Con el objetivo de que los menús emergentes sean compatibles con Android 2.x y Android 4.x se desarrollan en un entorno de compatibilidad y se agrega la librería Android Support Library al proyecto para lograr el resultado esperado. De esta manera, al ejecutar la aplicación en Android 2.x se ve como aparece en la Figura 18. Y en Android 4.x se muestra como aparece en la Figura 19.

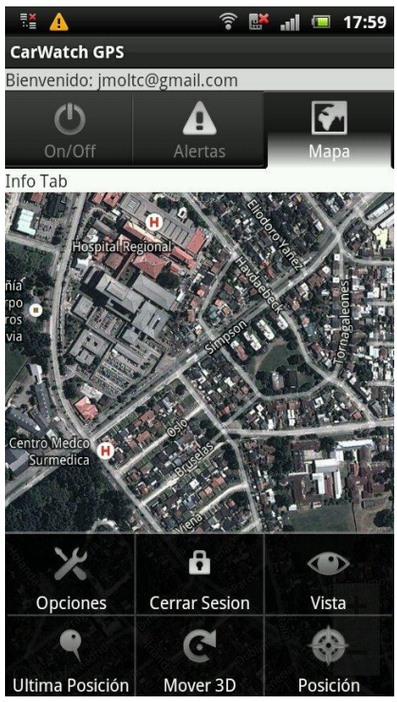

Figura 18. Menú Emergente en Android 2.x

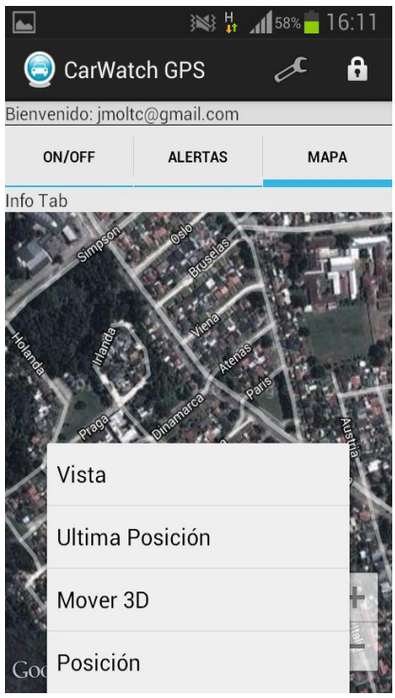

Figura 19. Menú Emergente en Android 4.x

5.3.5 Prueba de Concepto Nº 5 "Obtener localización del dispositivo emisor y almacenarla en la base de datos"

Para realizar esta prueba, se desarrolla una pequeña aplicación que obtiene la localización geográfica del dispositivo en base a las antenas de telefonía celular y el GPS del dispositivo. El objetivo de este experimento, es obtener la localización que está en coordenadas de latitud y longitud, y almacenarla en un servidor central. La aplicación desarrollada se muestra en la Figura 20.

Figura 20. Aplicación de prueba de localización.

La aplicación se encarga de actualizar la ubicación del dispositivo, y enviarla a través de Internet al servidor que se encuentra alojado en el Instituto de Informática. En la Figura 21 se aprecian los datos almacenados en la base de datos, con esto se cumple el objetivo del experimento.

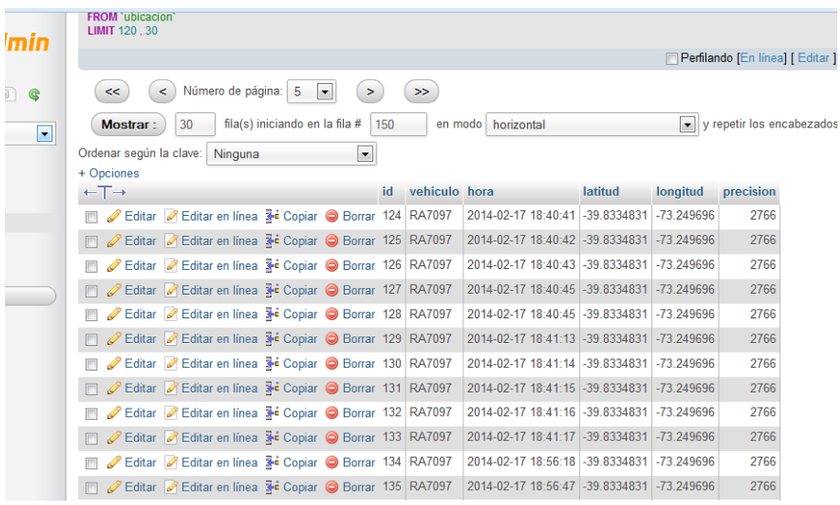

Figura 21. Localización almacenada en la base de datos.

5.3.6 Prueba de Concepto Nº 6 “Mostrar localización en el dispositivo cliente”

Para esta prueba se utiliza la API de Google para obtener los mapas y se obtiene la localización desde el Servidor del Instituto almacenada en la prueba anterior. Luego de un par de intentos se logra visualizar el mapa en la pantalla del dispositivo como se ve en la Figura 22.

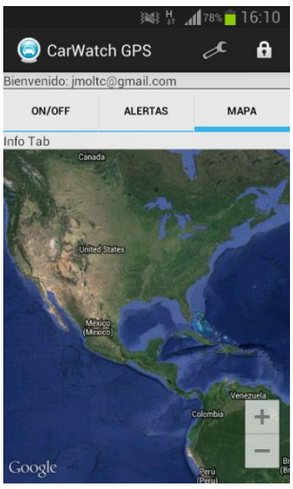

Figura 22. Despliegue de mapa con la API de Google.

Ahora se necesita obtener la ubicación almacenada en la base de datos del servidor, para esto se desarrolla un servicio web que atiende al dispositivo cuando éste necesita información del servidor. Con esto se puede comprobar en la Figura 23 que el objetivo del experimento se cumple, mostrando la localización en el mapa.

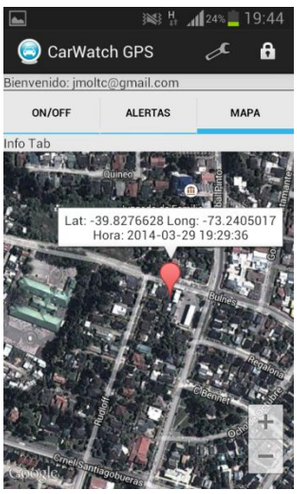

Figura 23. Localización mostrada en el mapa.

5.3.7 Prueba de Concepto Nº 7 “Sensor de presión”

Esta prueba consiste en conectar un sensor de presión al sistema a través de la placa IOIO. Para lo cual, utilizaremos un sensor de presión piezoresistivo modelo ZFLEX A201-100, marca *Tekscan* como el mostrado en la Figura 24.

Figura 24. Sensor de presión ZFLEX A201-100 de *Tekscan*[11]

El circuito con el sensor de presión, se muestra en la Figura 25. Donde se conecta a través de la placa IOIO al sistema, para lograr levantar un evento cada vez que el sensor sea presionado fuera de los rangos normales de reposo.

[11] Imagen obtenida desde https://www.sparkfun.com/products/8685

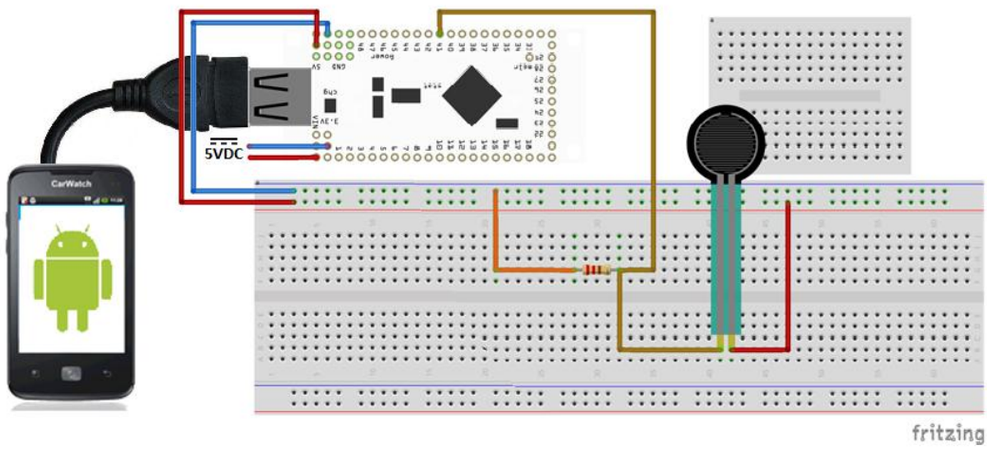

Figura 25. Circuito del sensor de presión.

En la Figura 26, se muestra una aplicación de prueba, que muestra la presión ejercida sobre el sensor de presión en tiempo real, donde podemos ver que los datos son capturados por el sistema cuando el sensor es presionado, y de esta forma pueden ser tratados y enviados como una alerta al dispositivo cliente.

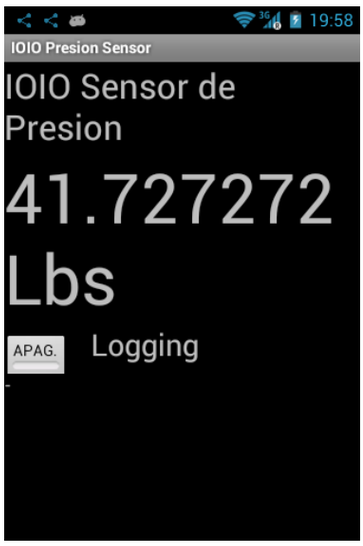

Figura 26. Aplicación de prueba para el sensor de presión.

5.4 Análisis y Diseño

5.4.1 Diagrama General de Casos de Uso

En la Figura 27, se muestra el diagrama general de casos de uso.

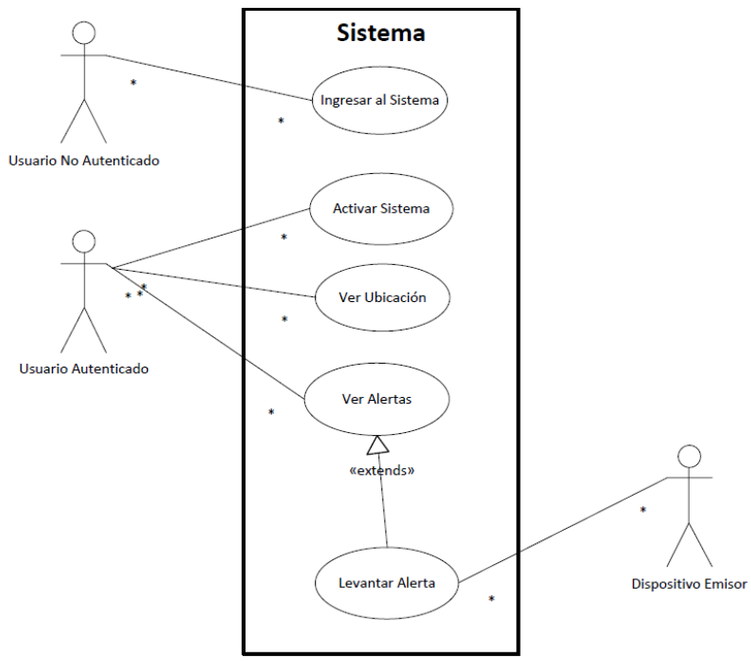

Figura 27. Diagrama general de casos de uso.

5.4.2 Casos de Uso del Sistema

5.4.2.1 Ingresar al Sistema

Caso de Uso: Ingresar al Sistema

Actores: Usuario No Autenticado

Propósito: Validar su identidad con sus credenciales.

Resumen: Un usuario no autenticado, accede a la aplicación y se despliega una pantalla para ingresar su nombre de usuario y contraseña. El usuario ingresa sus credenciales y se valida el acceso.

Requisitos Funcionales: R0

En la Tabla 4 se muestra el curso normal de los eventos y en la Figura 28 se muestra el diagrama de pantalla del caso de uso real.

Tabla 4. Curso normal de los eventos CU Ingresar al Sistema

Actor	Respuesta del Sistema
1. El usuario no autenticado accede a la aplicación.	
	2. El sistema despliega pantalla para ingresar usuario y contraseña.
3. El usuario ingresa su nombre de usuario (1) y contraseña (2) y presiona en Ingresar (3).	
	4. El sistema cambia a la pantalla inicial del usuario autenticado.

Figura 28. Diagrama de pantalla CU Ingresar al Sistema.

5.4.2.2 Activar Sistema

Caso de Uso: Activar Sistema

Actores: Usuario

Propósito: Armar sistema de alarma del vehículo

Precondición: El usuario debe estar autenticado.

Resumen: El usuario accede a la pantalla de activación del sistema para armar la alarma del vehículo.

Requisitos Funcionales: R5

En la Tabla 5 se muestra el curso normal de los eventos y en la Figura 29 se muestra el diagrama de pantalla del caso de uso real.

Tabla 5. Curso normal de los eventos CU Activar Sistema

Actor	Respuesta del Sistema
1. El caso de uso comienza cuando el usuario quiere armar el sistema de alarma del vehículo y accede a la pantalla de activación.	
2. El usuario presiona en Activar (1).	
	3. El sistema despliega el mensaje Sistema Ok, indicando que el sistema se ha activado correctamente.

Figura 29. Diagrama de pantalla CU Activar Sistema.

5.4.2.3 Ver Ubicación

Caso de Uso: Ver Ubicación

Actores: Usuario

Propósito: Ver ubicación actual del vehículo

Precondición: El usuario debe estar autenticado.

Resumen: El usuario accede a la pantalla del mapa y se despliega la última ubicación conocida.

Requisitos Funcionales: R4, R7

En la Tabla 6 se muestra el curso normal de los eventos y en la Figura 30 se muestra el diagrama de pantalla del caso de uso real.

Tabla 6. Curso normal de los eventos CU Ver Ubicación

Actor	Respuesta del Sistema
1. Este caso de uso comienza cuando el usuario quiere conocer la ubicación actual del vehículo y accede a la pantalla del mapa (1).	
	2. El sistema despliega el mapa y centra la ubicación indicando las coordenadas y la hora actual.

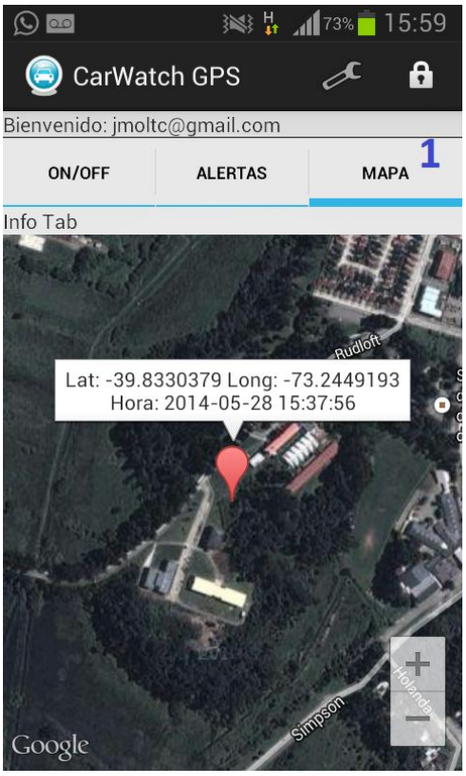

Figura 30. Diagrama de pantalla CU Ver Ubicación.

5.4.2.4 Ver Alertas

Caso de Uso: Ver Alertas

Actores: Usuario

Propósito: Ver hora de activación de la alarma.

Precondición: El usuario debe estar autenticado.

Resumen: El usuario accede a la pantalla de alertas para ver hora de activación de la alarma.

Requisitos Funcionales: R3, R4, R6

En la Tabla 7 se muestra el curso normal de los eventos y en la Figura 31 se muestra el diagrama de pantalla del caso de uso real.

Tabla 7. Curso normal de los eventos CU Ver Alertas

Actor	Respuesta del Sistema
1. Este caso de uso comienza cuando el usuario quiere ver el historial de alertas del sistema y accede a la pantalla de alertas (1).	
	2. El sistema despliega el histórico de alertas con fecha y hora.

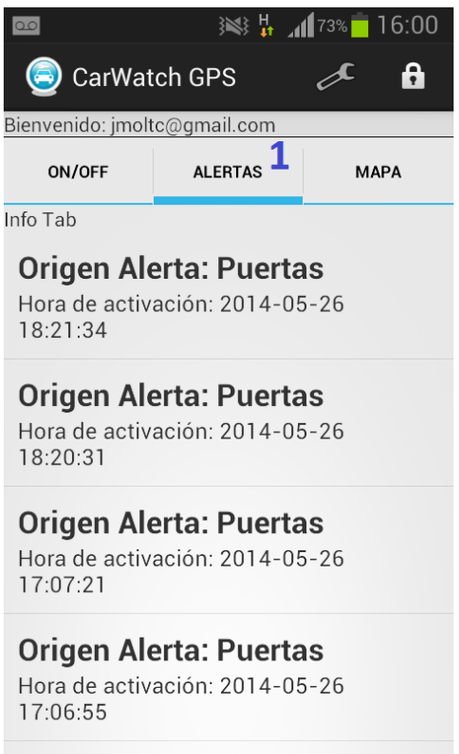

Figura 31. Diagrama de pantalla CU Ver Alertas.

5.4.2.5 Levantar Alerta

Caso de Uso: Levantar Alerta

Actores: Dispositivo Emisor, Usuario

Propósito: Enviar una alerta al dispositivo del usuario.

Precondición: El sistema de alertas debe estar armado.

Resumen: El sistema envía una alerta de activación de alarma al usuario.

Requisitos Funcionales: R1, R2, R3, R4

En la Tabla 8 se muestra el curso normal de los eventos y en la Tabla 9 se muestra un curso alternativo. En la Figura 32 se muestra un ejemplo de alerta recibida por el dispositivo del usuario.

Tabla 8. Curso normal de los eventos CU Levantar Alerta.

Actor	Respuesta del Sistema
1. Este caso de uso comienza cuando el dispositivo emisor levanta una alerta de activación de alarma.	
	2. El sistema despliega una alerta visual y auditiva en el dispositivo del usuario.

Tabla 9. Curso alternativo de los eventos CU Levantar Alerta

Actor	Respuesta del Sistema
1. Este caso de uso comienza cuando el dispositivo emisor levanta una alerta de activación de alarma.	
	2. El sistema intenta desplegar una alerta en el dispositivo del usuario, pero este se encuentra sin conectividad de internet.
3. El dispositivo emisor envía un SMS de alerta al dispositivo del usuario.	

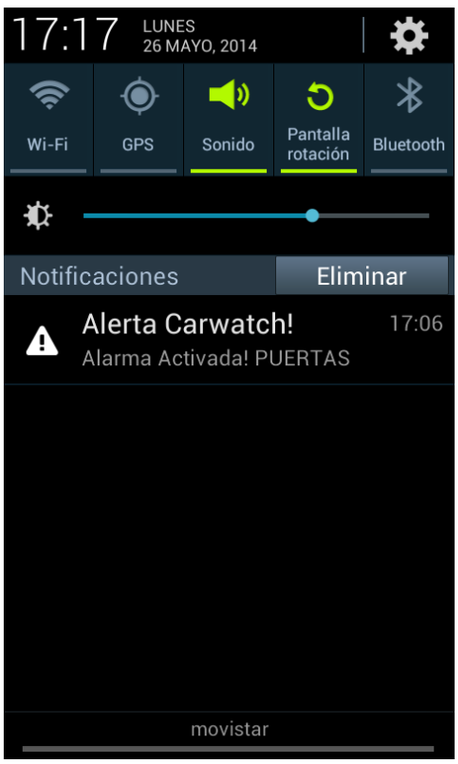

Figura 32. Ejemplo de alerta en dispositivo del usuario.

El caso de uso Levantar Alerta puede extender al CU Ver Alertas. En la eventualidad de que el usuario vea oportunamente la alerta enviada al dispositivo se continúa con el caso de uso Ver Alertas.

5.4.3 Diagramas de Secuencia

Los diagramas de secuencia de las Figuras 33, 34 y 35 muestran gráficamente los eventos que fluyen desde el usuario del sistema hacia el servidor central.

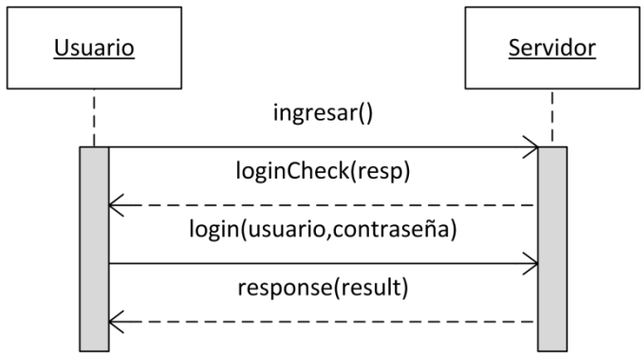

Figura 33. Diagrama de secuencia para ingresar al sistema.

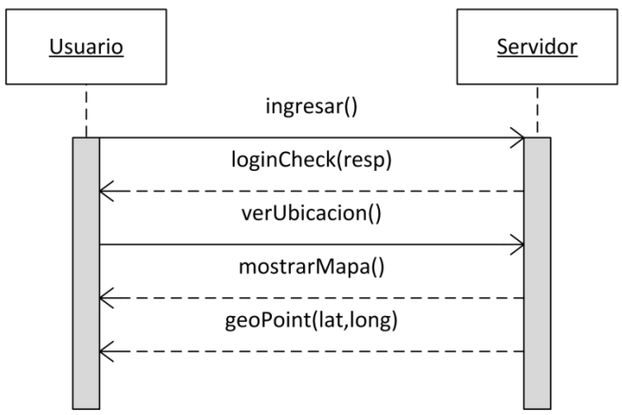

Figura 34. Diagrama de secuencia para ver ubicación.

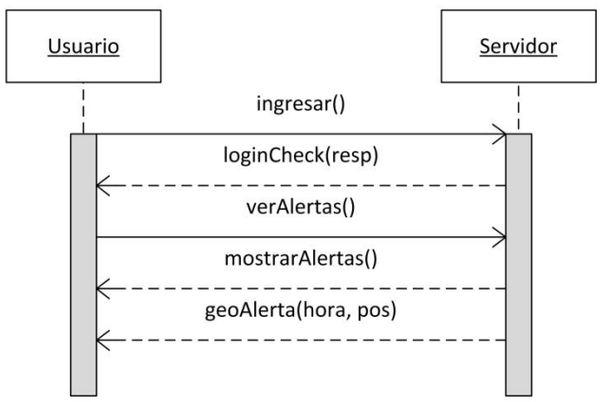

Figura 35. Diagrama de secuencia para ver alertas.

El diagrama de secuencia de la Figura 36, muestra la comunicación entre usuario, servidor y el dispositivo emisor para activar el sistema.

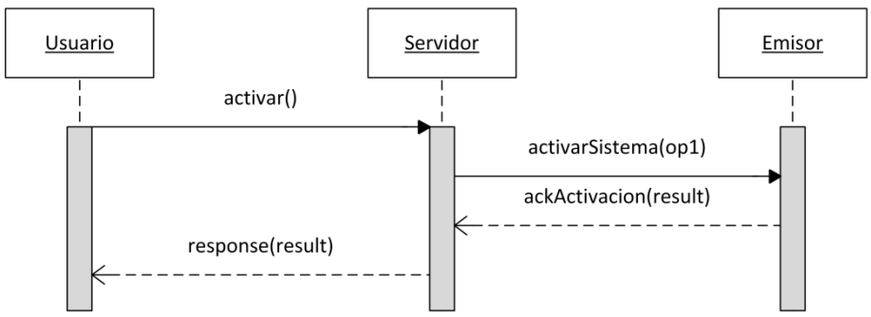

Figura 36. Diagrama de secuencia para activar el sistema.

El diagrama de secuencia de la Figura 37 muestra cómo se levanta una alerta desde el dispositivo emisor, posteriormente se comunica con el servidor y alerta con un mensaje al usuario.

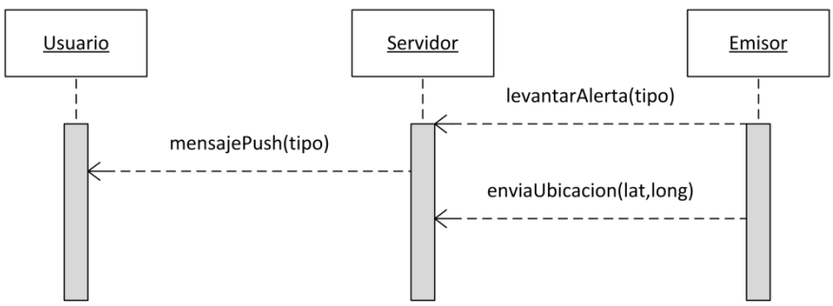

Figura 37. Diagrama de secuencia levantar alerta.

5.4.4 Diagrama de Clases

En la Figura 38 se muestra el diagrama de clases de la aplicación del dispositivo emisor.
En la Figura 39 se muestra el diagrama de clases de la aplicación del cliente.

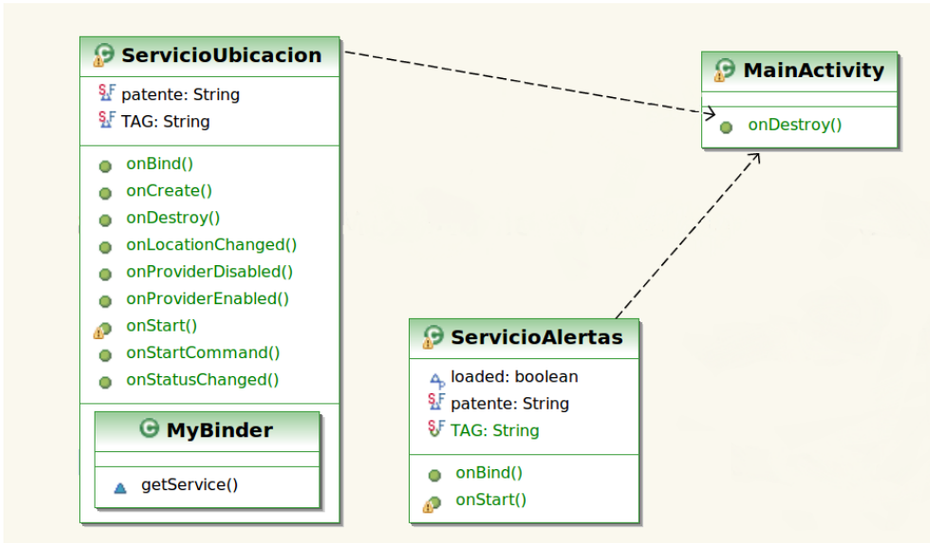

Figura 38. Diagrama de clases aplicación emisor.

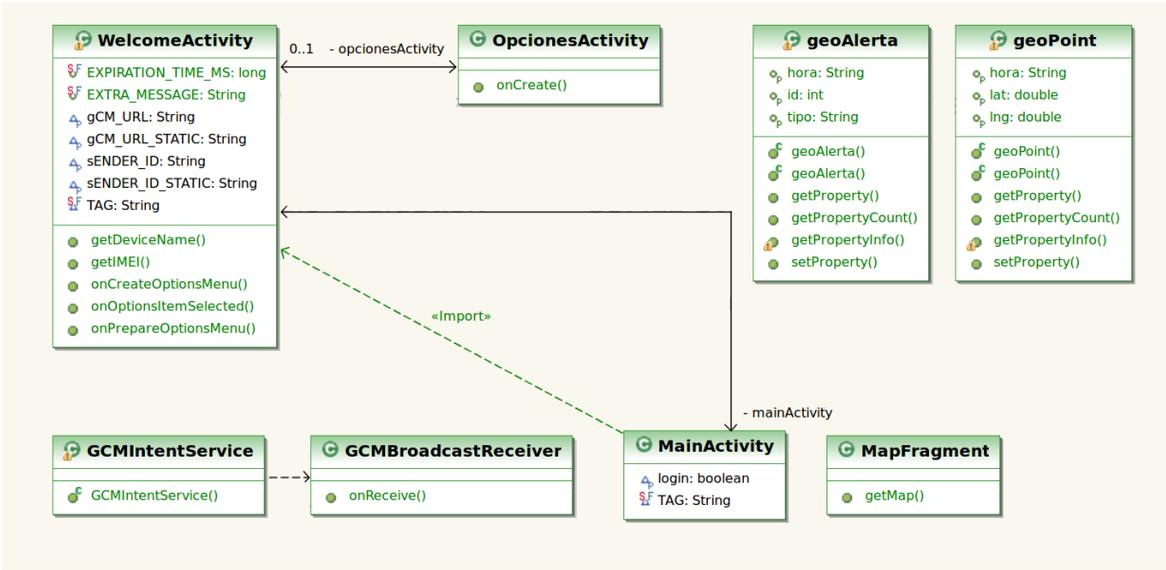

Figura 39. Diagrama de clases aplicación receptor/cliente.

5.4.5 Modelo de Base de Datos

En la Figura 40 se muestra el modelo de base de datos del servidor.

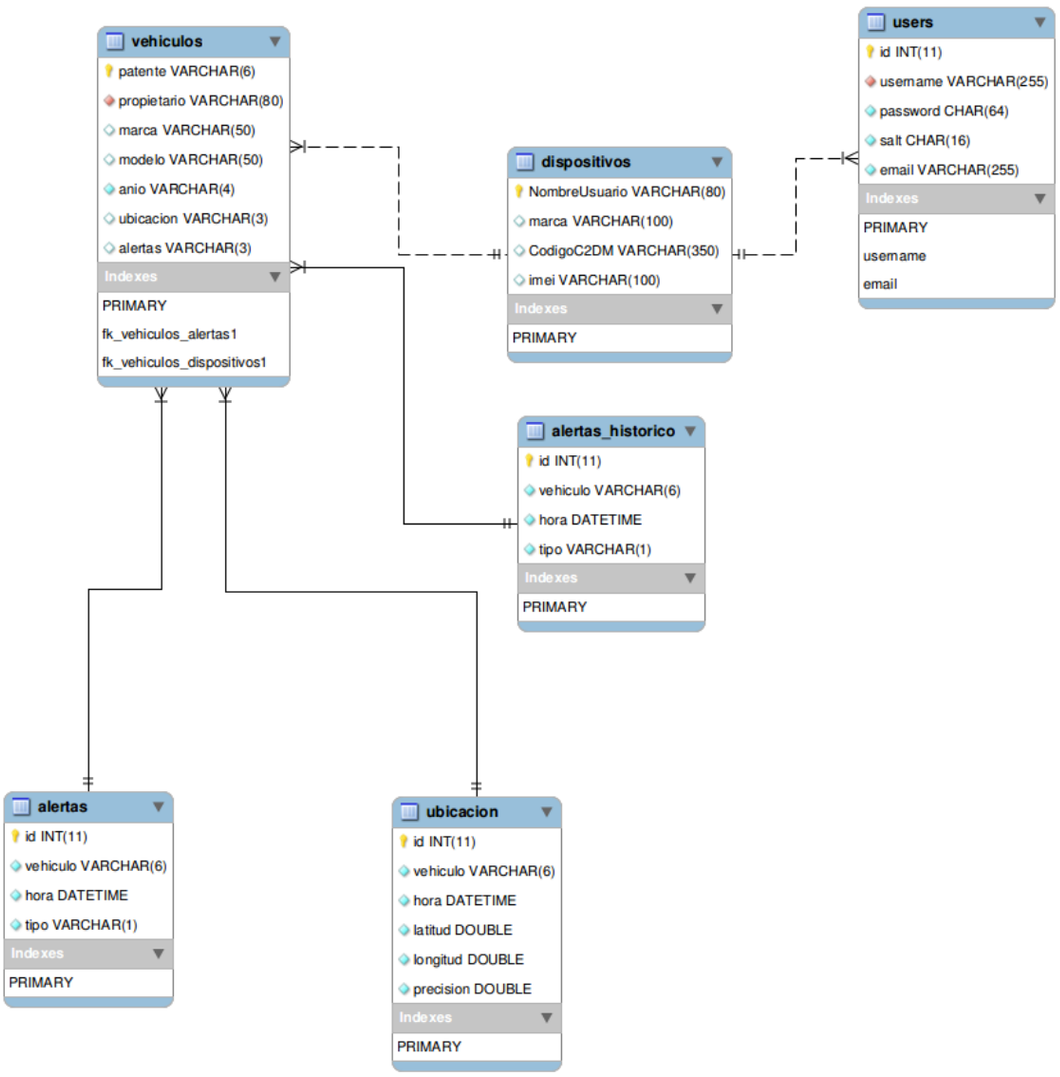

Figura 40. Modelo de base de datos del servidor.

5.5 Solución final

5.5.1 Uso del sistema

El presente proyecto de titulación tiene como producto, además de todas las interfaces de comunicación, dispositivos externos y el servidor de bases de datos instalado en el instituto, los cuales son necesarios para el funcionamiento del sistema, una aplicación móvil desarrollada para el proyecto, la que recibe el nombre de CarWatch GPS. Para hacer uso de la aplicación, ésta es instalada en el dispositivo móvil del usuario, posteriormente se debe buscar la aplicación en el teléfono móvil, la cual aparece con un ícono azul con un logotipo de un automóvil con una llave en la parte superior, bajo el nombre de CarWatch GPS como se aprecia en la Figura 41.

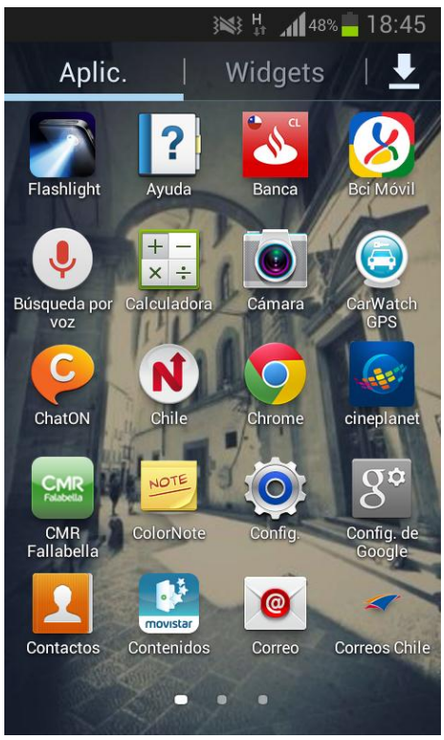

Figura 41. Aplicaciones instaladas en el dispositivo.

Al ejecutar la aplicación por primera vez el usuario debe ingresar su correo electrónico y contraseña como se muestra en la Figura 42, previamente el usuario debe estar registrado en www.car-watch.net y poseer vehículos registrados bajo su usuario.

Figura 42. Pantalla de *login*.

El sistema asocia los vehículos de cada usuario por su correo electrónico. Cuando el cliente ya tiene un vehículo asociado en sistema, solo tiene que autenticarse con sus credenciales, con esto el dispositivo cliente queda en línea y monitoreando.

5.5.2 Sistema final del dispositivo emisor

En la Figura 43 se muestra el esquema de conexión final del prototipo.

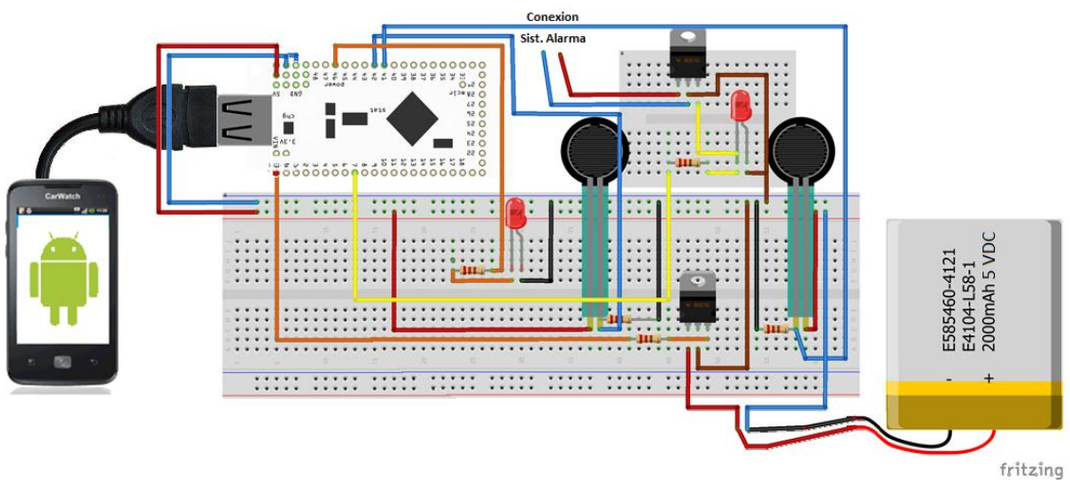

Figura 43. Esquema del cableado final.

En el esquema de cableado final, se conecta el dispositivo móvil con el software desarrollado para el emisor, encargado de alertar al servidor central en caso que se active

el sistema de alarma del automóvil. Además es el encargado de enviar al servidor central la ubicación geográfica del automóvil. El sistema está alimentado por una batería de 5V de hasta 1 Ampere/hora. La cual es recargada eventualmente con el sistema eléctrico del vehículo, y esta a su vez, carga la batería del dispositivo móvil. Entre los LED indicadores y el sistema de alarmas, se ubican dos reguladores de voltaje, para proteger los circuitos de la placa IOIO frente al flujo de corriente del vehículo.

En la figura 44 se muestra el prototipo instalado en el automóvil.

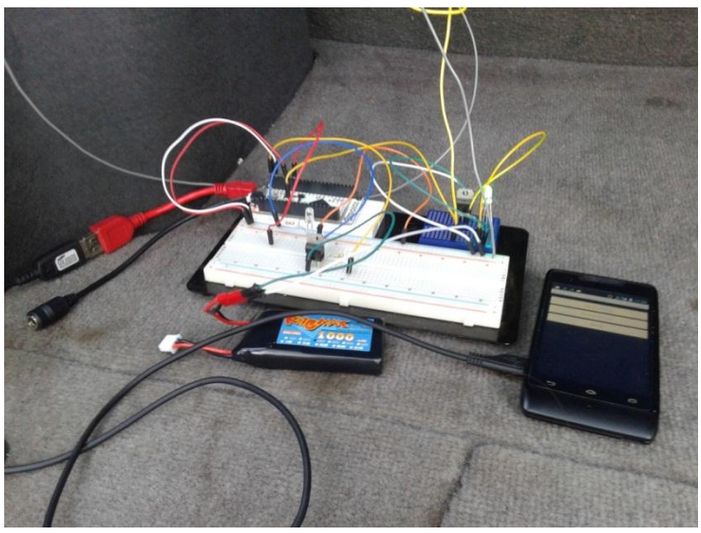

Figura 44. Prototipo instalado en el automóvil.

5.5.3 Capturas del software en el dispositivo cliente

Cuando se activa alguna alerta del sistema, por ejemplo, cuando se abre una puerta del vehículo el sistema envía una alerta al dispositivo del cliente. En la Figura 45 se muestra el sistema de alertas y en la Figura 46 se muestra el mapa con la última localización conocida.

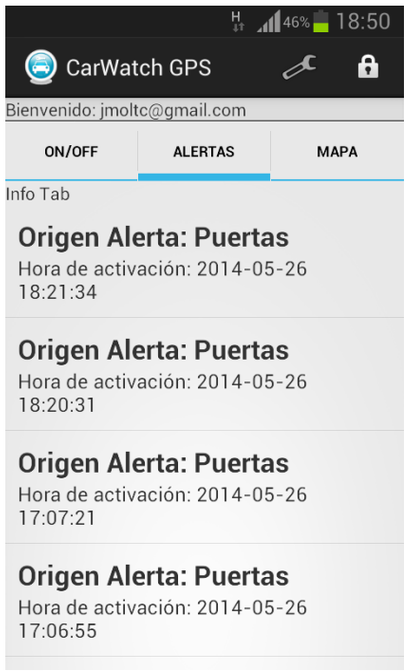

Figura 45. Pantalla del sistema de alertas.

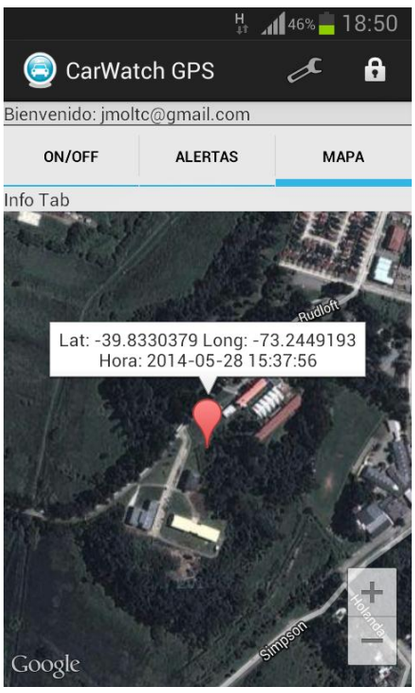

Figura 46. Localización geográfica del automóvil.

En el apartado de opciones, se puede cambiar el servidor de notificaciones, en caso de que el sistema vaya a ser utilizado con un servidor distinto al del Instituto de Informática. El menú de opciones, se muestra en la Figura 47.

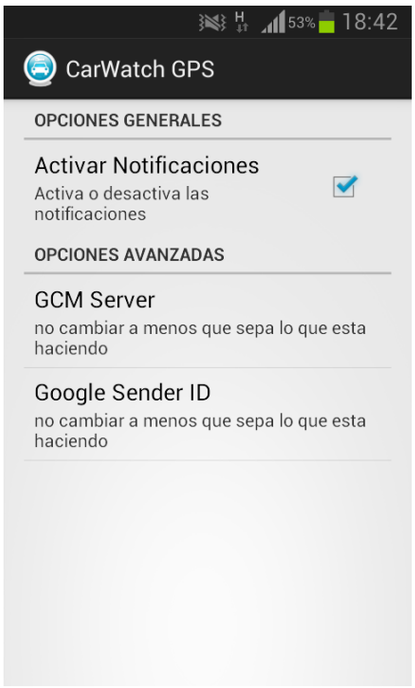

Figura 47. Menú de opciones de la aplicación.

5.5.4 Sistema de consulta vía web

Además de la aplicación móvil CarWatch GPS, el sistema desarrollado incorpora un módulo de consulta vía web, accesible desde la dirección web www.car-watch.net , donde se puede consultar el registro histórico de activaciones del sistema y la ubicación detallada del vehículo. En la Figura 48 se muestra el registro histórico de activaciones del sistema y en la Figura 49 se muestra la ubicación del vehículo en el mapa.

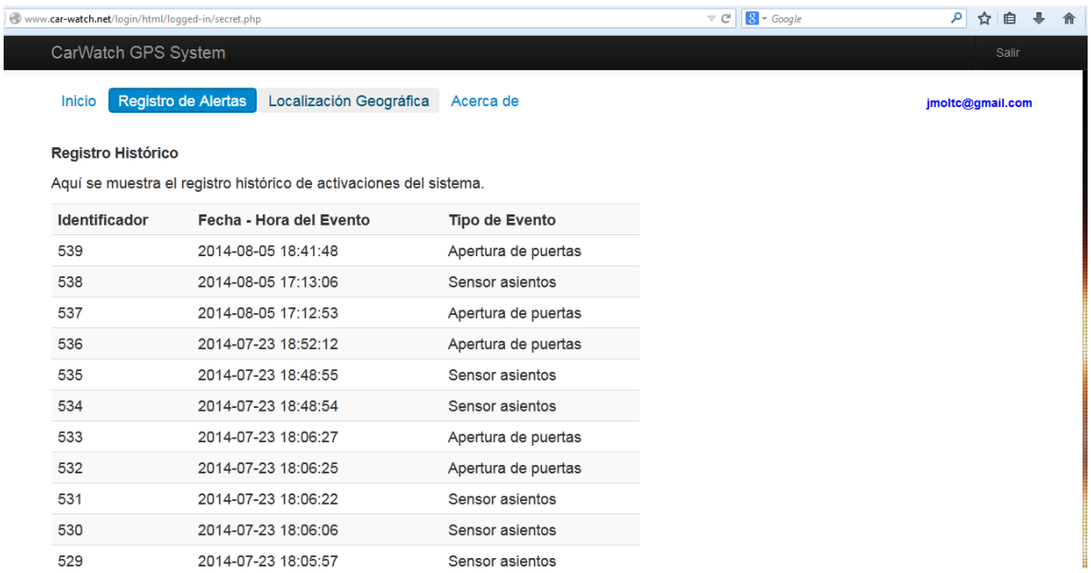

Figura 48. Registro histórico de activaciones del sistema de alarma.

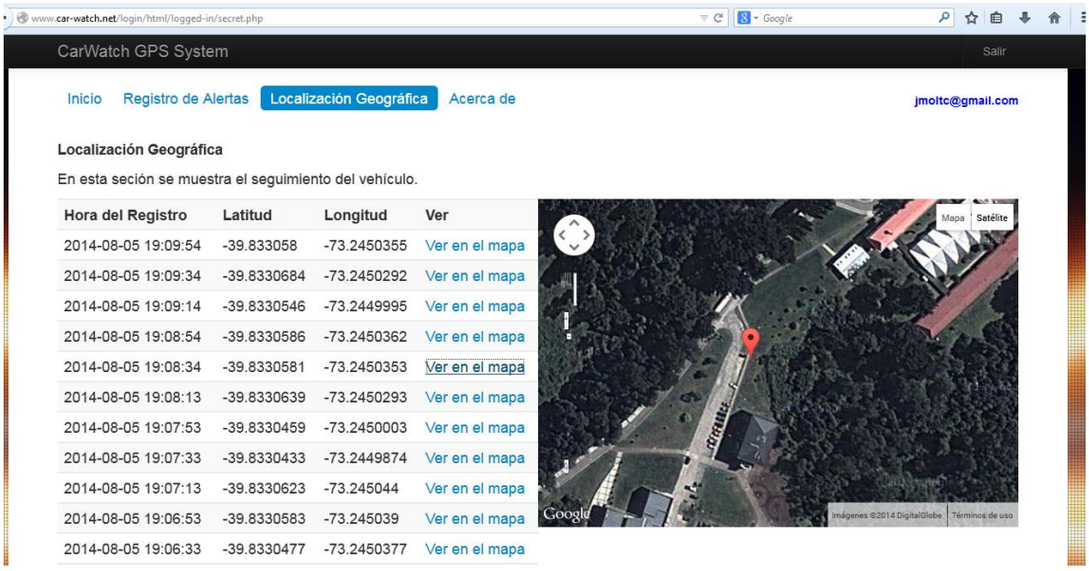

Figura 49. Ubicación del vehículo en el mapa.

De esta forma, el usuario final puede consultar la ubicación actual del vehículo, como también el registro histórico de lugares en los que estuvo anteriormente. También se puede obtener información acerca de las activaciones del sistema de alarmas, mostrando todas las activaciones, tanto de los sensores de puertas como los incorporados en los asientos.

6. CONCLUSIONES

6.1 Conclusiones

El desarrollo de aplicaciones móviles siempre se ve como una interfaz de aplicación para un usuario final y se olvidan las diferentes interfaces, servidores y servicios que deben interactuar en una arquitectura MCC (*Mobile Cloud Computing*) para que el software instalado en el dispositivo cliente cumpla correctamente su función.

En el desarrollo de este proyecto se logra un amplio conocimiento de los distintos componentes que conforman un sistema con una red de sensores, servicios web, redes de celulares, comunicación remota de los sensores con los servidores, además de la interacción entre servidores y dispositivos cliente.

Se determina utilizar una arquitectura MCC, ya que en esta arquitectura se tiene la capacidad de abstracción de ver a los dispositivos conectados a la nube como parte del mismo sistema. Se puede lograr que un dispositivo pueda asignar tareas a otros dispositivos móviles, como si fueran recursos de la nube. Además cada dispositivo puede consumir recursos provistos por la nube directamente.

Por otra parte, el uso de la placa IOIO-OTG colabora directamente con la arquitectura del sistema, ya que la placa IOIO-OTG trabaja como una extensión del mismo teléfono, y permite operar al sistema en conjunto, como si se tratara de un solo dispositivo.

También se evidencia que gracias a la capacidad de cómputo de estos pequeños dispositivos móviles, pueden ser utilizados en la solución de problemáticas de diversos ámbitos tales como el monitoreo de sistemas productivos, monitoreo del medio ambiente, cuidado de la salud, hogares inteligentes, y también, por qué no, en la nueva visión tecnológica-social de las ciudades inteligentes.

6.2 Trabajo futuro

El prototipo presentado en este proyecto deja varias aristas de trabajo e investigación. Se pueden embeber más funcionalidades al prototipo que no fueron integradas en este proyecto, en la sección 3.1 del documento, se detallan algunos sensores que no fueron incorporados a este prototipo, ya que debido a razones de presupuesto y el acotado

tiempo de entrega del proyecto, no fue posible incorporarlos. Además, el prototipo implementado, se puede utilizar como base para otros tipos de sistema de monitoreo, como conocer el estado de maquinaria industrial, conocer la ubicación de activos importantes para ciertas instituciones, o también puede aplicarse a otras áreas, como el cuidado y monitoreo de personas de la tercera edad, monitoreo de bebés o el cuidado de personas con capacidades reducidas.

REFERENCIAS

[Coo13] Cooperativa (2013). Disponible en
http://www.cooperativa.cl/noticias/pais/policial/robo-de-vehiculos/carabineros-mas-de-32-mil-autos-fueron-robados-en-2012/2013-02-08/182946.html
Consultado el 4 de abril de 2013.

[Ent12] Área de Prensa Entel (2012). Disponible en
http://smartphones.entel.cl/smartphones/posts/venta-de-smartphones-en-chile-supera-los-equipos-convencionales
Consultado el 20 de abril de 2013.

[Gee09] Geeknizer. REST vs. SOAP - The Right WebService. Disponible en
http://geeknizer.com/rest-vs-soap-using-http-choosing-the-right-webservice-protocol/
Consultado el 11 de abril de 2014.

[Lib13] Libertad digital, (2013). Android, imparable: 59% de cuota de mercado en '*smartphones*'. Disponible en http://www.libertaddigital.com/internet/2012-05-26/android-imparable-59-de-cuota-de-mercado-en-smartphones-1276459645/
Consultado el 13 de julio de 2013.

[Meg12] Megazine (2012). Tipos de alarmas de automóviles y sistemas de seguridad en vehículos. Disponible en http://megazine.co/varios-tipos-de-alarmas-de-automóviles-y-sistemas-de-vehículos-seguridad_54906.html
Consultado el 18 de julio de 2013.

[Net13] Netcraft.(2013) August Web Server Survey. Disponible en
http://news.netcraft.com/archives/2013/08/09/august-2013-web-server-survey.html
Consultado el 2 de abril de 2014.

[Oli13] Olimex Chile, (2013). IOIO para Android. Disponible en
http://www.olimex.cl/product_info.php?products_id=854
Consultado el 3 de julio de 2013.

[Ora14] Oracle. MySQL Specifications. Disponible en
http://www.oracle.com/es/products/mysql/index.html?ssSourceSiteId=null
Consultado el 2 de abril de 2014.

[Per] Pergamino Virtual. (n.d.) Servidor (*def*). Disponible en
http://www.pergaminovirtual.com.ar/definicion/Servidor.html
Consultado el 4 de abril de 2014.

[Php] PHP.net ¿Qué es PHP? (n.d). PHP (*def*) Disponible en
http://www.php.net/manual/es/intro-whatis.php
Consultado el 17 de abril de 2014.

[Red13] RedInnovaCom (2013). Sistemas Operativos. Disponible en http://www.redinnovacom.es/index.php?option=com_k2&view=item&id=196:sistemas-operativos
Consultado el 10 de Julio de 2013.

[Som09] Ian Sommerville. (2009). Ingeniería del Software. 7ª Edición. Pearson. Addison Wesley. 62-64.

[Spa13] SparkFun Electronics (2013). Disponible en https://www.sparkfun.com/products/10585
Consultado el 10 de abril de 2013.

[Uni13] Univision (2013). "Android el sistema operativo más usado en el mundo". Disponible en http://noticias.univision.com/article/1738822/2013-11-13/tecnologia/noticias/android-es-el-sistema-operativo-movil-mas-usado-en-el-mundo
Consultado el 7 de abril de 2014.

[Upv13] Universidad Politécnica de Valencia (2013). Disponible en http://www.upv.es/satelite/trabajos/pract_15/tecno.htm
Consultado el 19 de mayo de 2013.

[W3C14] W3C España (2014). Servicios Web. Disponible en http://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb
Consultado el 11 de abril de 2014.

[Xat14] Xataka Android (2014). "Gartner prevé que se superarán los 2.000 millones de dispositivos Android en 2014". Disponible en http://www.xatakandroid.com/mercado/gartner-preve-que-se-superaran-los-2-mil-millones-de-dispositivos-android-en-2014
Consultado el 11 de abril de 2014.

ANEXOS

ANEXO A: Código de ejemplo IOIO

```
public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.main);
      private ToggleButton boton1 = (ToggleButton)
findViewById(R.id.boton1);
}

class Looper extends BaseIOIOLooper {

      private DigitalOutput led;

      @Override
      protected void setup() throws ConnectionLostException {
            led = ioio_.openDigitalOutput(46, false);
      }

      @Override
      public void loop() throws ConnectionLostException {
            led.write(boton1_.isChecked());
            try {
                  Thread.sleep(100);
            }

            catch (InterruptedException e) {
            }
      }
}
```

ANEXO B: Código servidor Push

B.1 Código de ejemplo servidor Push en .NET

```
public class ServicioRegistroGCM : System.Web.Services.WebService
{

[WebMethod]
public int RegistroCliente(string usuario, string regGCM)
{
   SqlConnection con =
      new SqlConnection( @"Data Source=MOLT-PC\SQLEXPRESS;Initial
Catalog=DBUSUARIOS;Integrated Security=True");

   con.Open();
   string cod = CodigoCliente(usuario);
   int res = 0;
   string sql = "";

   if (cod == null)
   sql = "INSERT INTO Usuarios (NombreUsuario, CodigoC2DM) VALUES
(@usuario, @codigo)";

   else
   sql = "UPDATE Usuarios SET CodigoC2DM = @codigo WHERE NombreUsuario
= @usuario";

SqlCommand cmd = new SqlCommand(sql, con);
cmd.Parameters
     .Add("@usuario", System.Data.SqlDbType.NVarChar).Value = usuario;
            cmd.Parameters
     .Add("@codigo", System.Data.SqlDbType.NVarChar).Value = regGCM;

            res = cmd.ExecuteNonQuery();
            con.Close();
            return res;
}
```

```
[WebMethod]
public string CodigoCliente(string usuario)
{
    SqlConnection con =
    new SqlConnection(
    @"Data Source=SGOLIVER-PC\SQLEXPRESS;Initial
Catalog=DBUSUARIOS;Integrated Security=True");

            con.Open();
            string sql = "SELECT CodigoC2DM FROM Usuarios WHERE
NombreUsuario = @usuario";

            SqlCommand cmd = new SqlCommand(sql, con);

            cmd.Parameters.Add("@usuario",
System.Data.SqlDbType.NVarChar).Value = usuario;

            string cod = (String)cmd.ExecuteScalar();
            con.Close();
            return cod;
        }
}}
```

B.2 Código servidor Push portado a PHP

```
function RegistroCliente($usuario, $regGCM, $model, $imei)

{
$link = mysql_connect(DBURL, DBUSER, DBPASS)
or die('No se pudo conectar: ' . mysql_error());

mysql_select_db(DBNAME)
or die('No se pudo seleccionar la base de datos');

mysql_query('SET CHARACTER SET utf8');

$resp = CodigoCliente($usuario);

if ($resp === NULL)
{
      $query  = "INSERT INTO dispositivos (NombreUsuario, marca,
CodigoC2DM, imei) VALUES
('".$usuario."','".$model."','".$regGCM."','".$imei."')";
            }
else
{
      $query = "UPDATE dispositivos SET CodigoC2DM = '".$regGCM."',
marca = '".$model."', imei = '".$imei."' WHERE NombreUsuario =
'".$usuario."'";
            }
$result = mysql_query($query)
            or die('Consulta fallida: ' . mysql_error());

$valor = 0;

if (!$result)
{$valor=0;}

else
{$valor=1;}

return $valor;
}
```

```
function CodigoCliente($usuario)
{

$link = mysql_connect(DBURL, DBUSER, DBPASS)
          or die('No se pudo conectar: ' . mysql_error());

mysql_select_db(DBNAME)
      or die('No se pudo seleccionar la base de datos');

mysql_query('SET CHARACTER SET utf8');

$query = "SELECT CodigoC2DM FROM dispositivos WHERE NombreUsuario =
'".$usuario."'";

$result = mysql_query($query)
    or die('Consulta fallida: ' . mysql_error());

$filas = mysql_num_rows($result);
$valor = NULL;

if ($filas==0)
{
      $valor = NULL;
}
else
{
      while ($row=mysql_fetch_assoc($result))
      {
            $valor = $row['CodigoC2DM'];
      }
}

return $valor;

}
```

ANEXO C: Lista de abreviaciones

API: Application Programming Interface

CPU: Central Processing Unit

C2DM: Cloud 2 (to) Device Messaging

CU: Caso de Uso

DC: Direct Current (Corriente Directa)

EEUU: Estados Unidos

GIT: es un software de control de versiones diseñado por Linus Torvalds

GNU/GPL: General Public License o Licencia General de GNU

HTTP: Hypertext Transfer Protocol

IDE: Integrated Development Environment

iOS: Sistema operativo de Apple

IIS: Internet Information Services, es un servidor web de Microsoft

INE: Instituto Nacional de Estadísticas

IOIO: comúnmente nombrado yo-yo, es el nombre de una placa de desarrollo

IOIO-OTG: es la segunda versión de la placa de desarrollo IOIO

LED: light-emitting diode (diodo emisor de luz)

MCC: Mobile Cloud Computing

.NET: Lenguaje de programación e IDE para desarrollo de Microsoft

OS: Operating System

PHP: Hypertext Pre-processor, es un lenguaje de desarrollo

USB: Universal Serial Bus, popular conector estándar de periféricos.

SMS: Short Message Service (servicio de mensajes cortos), mensajería de celulares

SMTP: Simple Mail Transfer Protocol, protocolo de transferencia de correo electrónico

SQL: Structured Query Language, lenguaje de acceso a bases de datos

SOAP: Simple Object Access Protocol, es un protocolo de comunicación entre procesos

WSDL: Web Service Description Languaje, formato para describir servicios web

XML: eXtensible Markup Language