



# Universidad Austral de Chile

---

Facultad de Ciencias de la Ingeniería

Escuela de Ingeniería Civil en Informática

## **GESTIÓN DE TICKETS DE ESPERA PARA CENTROS DE ATENCIÓN, UTILIZANDO NOTIFICACIONES PUSH ANDROID.**

Proyecto para optar al título de  
**Ingeniero Civil en Informática**

PROFESOR PATROCINANTE:  
JORGE ANTONIO MORALES VILUGRÓN  
INGENIERO ELECTRÓNICO  
M.B.A., D.E.A. MICROCONTROLADORES

PROFESOR CO-PATROCINANTE:  
MARÍA ELIANA DE LA MAZA WERNER  
INGENIERO CIVIL EN INFORMÁTICA  
MAGÍSTER EN INFORMÁTICA EDUCATIVA

PROFESOR INFORMANTE:  
N.N.  
N.N.

**MIGUEL ESTEBAN ORELLANA CONCHA**

VALDIVIA - CHILE

2015

## **AGRADECIMIENTOS**

ÍNDICE

ÍNDICE. . . . .	I
ÍNDICE DE TABLAS . . . . .	II
ÍNDICE DE FIGURAS . . . . .	III
RESUMEN . . . . .	IV
ABSTRACT . . . . .	V
1 INTRODUCCIÓN . . . . .	1
1.1 Antecedentes . . . . .	2
1.2 Motivación . . . . .	4
1.3 Impactos . . . . .	4
1.4 Definición del Proyecto . . . . .	5
1.4.1 Objetivo general . . . . .	5
1.4.2 Objetivos específicos . . . . .	5
1.5 Organización del documento . . . . .	5
2 MARCO TEÓRICO . . . . .	7
2.1 Tecnologías Asociadas . . . . .	7
2.1.1 Android . . . . .	7
2.1.2 <i>Google Cloud Messaging</i> para <i>Android</i> . . . . .	8
2.1.2.1 Arquitectura . . . . .	9
2.1.2.2 Enviar Mensajes . . . . .	10
2.1.2.3 Recibir Mensajes . . . . .	10
2.1.3 Notificaciones Push . . . . .	10
2.1.4 Web Service . . . . .	12
2.1.5 MySQL . . . . .	12
2.2 Teoría de Colas . . . . .	13
2.3 Internet de las cosas . . . . .	14
2.3.1 Big Data . . . . .	15
2.4 Estado del Arte . . . . .	16
2.4.1 Banco de Venezuela, Cola Virtual . . . . .	16
2.4.2 WAVETEC, Sistema de Administración de Filas . . . . .	17
2.4.2.1 Reportes en tiempo real . . . . .	17
2.4.2.2 Reportes de sucursales . . . . .	17
2.4.3 Q-sige, Sistema Inteligente de Gestión de Esperas . . . . .	18
3 SOLUCIÓN PROPUESTA . . . . .	19
3.1 Problemática . . . . .	19
3.2 Presentación de la solución . . . . .	19
3.3 Descripción de la metodología . . . . .	20
3.4 Descripción de Tecnologías a utilizar . . . . .	21
3.4.1 Entorno de desarrollo . . . . .	21
3.4.2 Intercambio de datos . . . . .	22
3.4.3 Base de datos . . . . .	24
3.4.4 Librerías utilizadas . . . . .	24
3.4.5 Patrón MVC . . . . .	24
3.5 Arquitectura del sistema . . . . .	25
4 ANÁLISIS Y DISEÑO DEL PROTOTIPO DEL SISTEMA. . . . .	27
4.1 Metodología . . . . .	27
4.2 Plan de trabajo . . . . .	27
4.2.1 Estrategia de implementación . . . . .	28
4.3 Análisis . . . . .	29
4.3.1 Descripción de los actores del sistema . . . . .	29
4.3.2 Casos de Uso . . . . .	30
4.4 Especificación de requerimientos . . . . .	31
4.4.1 Requisitos Funcionales . . . . .	31
4.4.2 Requisitos No Funcionales . . . . .	32
4.4.2.1 Casos de Uso Extendido del sistema . . . . .	33
4.5 Diseño . . . . .	36
4.5.1 Casos de Uso Real . . . . .	36
4.5.2 Diagramas de Secuencia . . . . .	42

4.5.3	Diagrama de Clases . . . . .	44
4.5.4	Diagrama de Procesos . . . . .	45
4.5.5	Modelo de Datos . . . . .	46
4.5.5.1	Diccionario de datos . . . . .	46
5	IMPLEMENTACIÓN DEL PROTOTIPO . . . . .	50
5.1	Configuración ambiente de desarrollo y problemas enfrentados . . . . .	50
5.2	Presentación de prototipos . . . . .	52
5.2.1	Prototipo 1: Aplicación básica en android Studio . . . . .	52
5.2.2	Prototipo 2: Cliente y Servidor GCM . . . . .	57
5.2.2.1	Cliente . . . . .	57
5.2.2.2	Servidor . . . . .	62
5.2.3	Prototipo 3: Comunicar Android y MySQL . . . . .	64
5.3	Solución final . . . . .	67
5.3.1	Servidor central . . . . .	67
5.3.2	Monitor . . . . .	67
5.3.3	Aplicación Web . . . . .	68
5.3.4	Aplicación Móvil . . . . .	69
6	PRUEBAS . . . . .	72
6.1	Prueba de Aplicación web. . . . .	72
6.1.1	RFQ - 001 Iniciar sesión en aplicación web. . . . .	72
6.1.2	RFQ - 002 Cargar y mostrar servicios. . . . .	73
6.1.3	RFQ - 003 Cargar y mostrar módulos de atención. . . . .	73
6.1.4	RFQ - 004 Enviar notificaciones desde aplicación web a <i>smartphone</i> . . . . .	74
6.1.5	RFQ - 005 Generar reportes. . . . .	74
6.1.6	RFQ - 006 Mostrar últimos turnos en atención. . . . .	75
6.1.7	RFQ - 007 Monitor de atenciones. . . . .	76
6.2	Prueba de Aplicación móvil. . . . .	76
6.2.1	RFQ - 008 Mostrar servicios. . . . .	77
6.2.2	RFQ - 009 Seleccionar un servicio. . . . .	77
6.2.3	RFQ - 010 Ver turno de atención. . . . .	78
6.2.4	RFQ - 011 Ver notificaciones. . . . .	78
6.2.5	RFQ - 012 Mostrar notificaciones nuevas automáticamente. . . . .	79
6.2.6	RFQ - 013 Mostrar información de la aplicación. . . . .	79
7	CONCLUSIONES . . . . .	80
7.1	Conslusiones . . . . .	80
7.2	Trabajo futuro . . . . .	81
	BIBLIOGRAFÍA . . . . .	82
	ANEXOS . . . . .	86
	Anexo A: Traducción a español extracto Ley de consumo Brasileña (Ley nº 8.078/90). . . . .	86

# ÍNDICE DE TABLAS

TABLA	PÁGINA
1 Versiones de Android soportadas por Google en la actualidad [And15]. . . . .	8
2 Componentes de GCM . . . . .	9
3 Requisitos Funcionales. . . . .	32
4 Requisitos No Funcionales. . . . .	32
5 Caso de uso extendido “Agregar Servicios”. . . . .	33
6 Caso de uso extendido “Agregar Módulo Atención”. . . . .	34
7 Caso de uso extendido “Seleccionar Servicio”. . . . .	35
8 Caso de uso Real “Agregar Servicios”. . . . .	36
9 Caso de uso Real “Agregar Módulo Atención”. . . . .	38
10 Caso de uso Real “Seleccionar Servicio”. . . . .	40
11 Diccionario de datos tabla “dispositivos”. . . . .	47
12 Diccionario de datos tabla “mensajes”. . . . .	47
13 Diccionario de datos tabla “modulo”. . . . .	48
14 Diccionario de datos tabla “servicios”. . . . .	48
15 Diccionario de datos tabla “usuarios”. . . . .	48
16 Diccionario de datos tabla “modulo_mensajes”. . . . .	48
17 Diccionario de datos tabla “modulo_servicios”. . . . .	49
18 Herramientas, estado y versión en máquina de trabajo. . . . .	50

# ÍNDICE DE FIGURAS

FIGURA	PÁGINA	
1	Arquitectura de GCM [And14]. . . . .	9
2	Ejemplo de Notificación Push en Android. [Sgo11g]. . . . .	11
3	Flujo de datos entre una aplicación <i>Android</i> , <i>Web Service PHP</i> y MySQL. [Myb13]. . . . .	12
4	Internet de las cosas [The14]. . . . .	15
5	<b>Objeto:</b> conjunto desordenado de pares nombre/valor. . . . .	23
6	<b>Arreglo:</b> colección de valores separados por ',' (coma). . . . .	23
7	<b>Valor:</b> cadena de caracteres, o número, u objeto, o arreglo, o true, o false, o null. . . . .	24
8	Patrón MVC.[And12] . . . . .	25
9	Arquitectura del sistema. . . . .	26
10	Carta Gantt que exhibe las etapas de desarrollo del sistema. . . . .	28
11	Diagrama de Casos de Uso Módulo web. . . . .	30
12	Diagrama de Casos de Uso Aplicación Móvil. . . . .	31
13	<i>Mockup</i> Caso de Uso 1 (CU01). . . . .	37
14	<i>Mockup</i> Caso de Uso 2 (CU02). . . . .	39
15	<i>Mockup</i> Caso de Uso 3 (CU03). . . . .	41
16	Diagrama de Secuencia Caso de Uso Agregar Servicios. . . . .	42
17	Diagrama de Secuencia Caso de Uso Agregar Módulo Atención. . . . .	42
18	Diagrama de Secuencia Caso de Uso Seleccionar Servicio. . . . .	43
19	Diagrama de Clases Aplicación Móvil. . . . .	44
20	Diagrama de Proceso Aplicación Móvil. . . . .	45
21	Modelo de datos. . . . .	46
22	Iniciar Apache desde el terminal de OS X. . . . .	51
23	Servicio Apache funcionando correctamente. . . . .	51
24	Acceso a carpeta apache2 y edición archivo httpd.conf. . . . .	52
25	Habilitar PHP para Apache. . . . .	52
26	Reiniciando Apache. . . . .	52
27	Modo Depuración por USB activado en <i>Android</i> 4.4.4. . . . .	53
28	Ventana bienvenida <i>Android Studio</i> . . . . .	54
29	Elección de API mínima requerida. . . . .	55
30	Proyecto nuevo en <i>Android Studio</i> . . . . .	56
31	Primera aplicación móvil de prueba. . . . .	57
32	Crear nuevo proyecto en Google Developers Console. . . . .	58
33	Resumen proyecto. . . . .	58
34	Activación de API para proyecto. . . . .	58
35	Crear clave nueva. . . . .	59
36	Crear clave nueva clave de servidor. . . . .	59
37	Crear clave de servidor. . . . .	59
38	Resumen servidor. . . . .	60
39	Instalación <i>Google Play Services</i> . . . . .	60
40	Clases del proyecto. . . . .	61
41	Incluir el <i>PROJECT ID</i> en la aplicación. . . . .	61
42	Registrar dispositivo desde la aplicación. . . . .	61
43	Codigo html del Servidor. . . . .	62
44	Codigo clase gcm_engine.php. . . . .	62
45	Servidor. . . . .	63
46	Notificación <i>Push</i> recibida. . . . .	63
47	Visualizacion contenido de notificación. . . . .	64
48	Registrar dispositivo en MySQL. . . . .	65
49	Código de función registrar dispositivo. . . . .	65
50	Código clase asíncrona JSONMain. . . . .	65
51	Código de función registrarDispositivo() en clase FuncionesApp(). . . . .	65
52	Porción de código de servicio web. . . . .	66
53	Código de función guardarDispositivos() en clase DB_Functions.PHP. . . . .	67
54	Datos almacenados correctamente en MySQL. . . . .	67

55	Monitor del sistema. . . . .	68
56	Aplicación web. . . . .	69
57	Pantalla principal 'Obtener Turno', a la derecha, y 'Menú' de opciones, a la izquierda,. . .	70
58	Dialogo confirmación obtener turno. . . . .	70
59	Pantalla 'Turno Actual'. . . . .	71
60	Pantalla con Notificación Push, a la izquierda. Visualización de la Notificación a la derecha.	71
61	RFQ-001 Ventana Inicio sesión. . . . .	72
62	RFQ-001 Ventana principal aplicación web. . . . .	73
63	RFQ-002 Combobox con servicios. . . . .	73
64	RFQ-003 Combobox con módulos de atención. . . . .	74
65	RFQ-004 Botón para enviar notificaciones. . . . .	74
66	RFQ-004 Gráfico barra número de atenciones por servicio. . . . .	75
67	RFQ-004 Gráfico líneas número de atenciones al mes. . . . .	75
68	RFQ-006 Turno en atención correspondiente al Servicio D. . . . .	76
69	RFQ-007 Monitor muestra turno, módulo y servicio. . . . .	76
70	RFQ-008 Servicios disponibles. . . . .	77
71	RFQ-009 Selección de Servicio A. . . . .	77
72	RFQ-010 Turno de atención obtenido. . . . .	78
73	RFQ-011 Notificación recibida desde aplicación web. . . . .	78
74	RFQ-012 Notificación Push. . . . .	79
75	RFQ-013 Información del proyecto. . . . .	79

## **RESUMEN**

Este Proyecto de Titulación pretende mejorar el proceso de atención de los usuarios que acuden a una oficina para realizar un trámite, de tal modo de optimizar el tiempo de espera en las interminables filas de este servicio.

El objetivo general del proyecto, es el desarrollo de un sistema capaz de solicitar un número de atención a través de una aplicación para Smartphones con Sistema Operativo Android, y hacer el seguimiento vía Notificaciones Push de los tickets de espera para personas que estén en algún centro de atención, con el objeto de optimizar los tiempos de espera. Es decir, los usuarios podrían salir del recinto y realizar otros trámites y con ello hacer más eficiente su tiempo.

Actualmente, las filas de espera en los diferentes centros de atención cuentan con un ticket manual y los clientes deben permanecer al interior del recinto. Las nuevas tecnologías emergentes y de uso masivo, como los Smartphones, representan una oportunidad para incorporar tecnología a éste proceso de atención. Particularmente el diseño de un tablero que envíe notificaciones del estado de la fila al Smartphone del cliente. Ésta información es almacenada y es la base para generar diversos reportes y con ellos mejorar la gestión de los servicios.

En este proyecto se propone una solución utilizando tecnología de vanguardia, que es la plataforma Web y la utilización de Smartphones con Sistema Android para el aprovechamiento de la información obtenida y seguimiento de las colas de espera.

Finalmente, se define la mejor forma de implementar este sistema acompañada de una validación de esta implementación.



## **ABSTRACT**

This project aims to improve the process of care of users who go to an office to complete a procedure, thereby optimizing the wait time in the endless queue of this service.

The overall objective of the project is to develop a system capable of requesting a number of care through an app for smartphones with Android operating system, and tracing Push Notifications tickets waiting for people who are in some center care, in order to optimize waiting times. That is, users could leave the premises and perform other procedures and thereby streamline your time.

Currently, the queues at the various centers have a manual ticket and customers must remain inside the enclosure. Emerging new technologies and widespread use, such as Smartphones, represent an opportunity to incorporate this technology into care process. In particular the design of a board to send notifications to the state of the row customer Smartphone. This information is stored and is the basis for generating various reports and with them improve the management of services.

In this project, a solution using cutting edge technology, which is the Web platform and the use of Smartphones with Android system for the use of information obtained and monitoring of queues is proposed.

Finally, define the best way to implement this system accompanied by a validation of this implementation.

## 1. INTRODUCCIÓN

En los últimos años hemos sido espectadores del avance veloz con que se han desarrollado las tecnologías de la información y telecomunicaciones. En este ámbito, las tecnologías móviles e internet han sido los servicios de mayor crecimiento en nuestro país[Pew14]. Los dispositivos móviles se han acercado y se han hecho parte de la vida diaria de las personas, buscando marcar una diferencia en los quehaceres cotidianos y claramente influir en una mejor calidad de vida para los usuarios.

En el presente, las prestaciones que ofrecen estos dispositivos de bolsillo superan ampliamente a los computadores de escritorio de inicios del siglo XXI. Además, se encuentran comunicados a Internet mediante diversas tecnologías inalámbricas. Esta comunicación no sólo permite a las personas mantener el contacto con su vida, trabajo, amistades; sino también permite que estos dispositivos se conecten con otros de forma independiente, realizando tareas en segundo plano para brindar más información de interés a las personas o empresas.

Actualmente, existe una alta interconexión digital de objetos cotidianos con Internet como: televisores, refrigeradores, automóviles incluso un termostato inteligente que regula la temperatura de una casa de forma remota[Gre13]. Lo anterior, abre paso a un nuevo paradigma donde un dispositivo de cualquier naturaleza es capaz de conectarse a Internet, siendo éste llamado el “Internet de las cosas” (IoT, *the Internet of Things*). Este nuevo concepto nos brinda no sólo el beneficio de tener todo el control desde nuestro smartphone; va más allá. Se trata de que al estar todo conectado podemos obtener una serie de información con la que antes no contabamos y, con la que podemos tomar mejores decisiones en diversos ámbitos. Por ejemplo, aplicaciones móviles que permitan distribuir mejor nuestro tiempo cuando realizamos varios trámites.

Es un hecho que en las principales provincias de Chile es muy elevada la concurrencia de personas que se encuentran en un centro de atención. En la actualidad, gracias a los avances en las tecnologías de información y telecomunicaciones, no es necesario realizar trámites personalmente en: pago de cuentas, transferencias bancarias, compra de productos, etc., ya que éstas se pueden realizar a través de aplicaciones Web. En cambio, hay otros que inevitablemente deben contar con la presencia del cliente.

Considerando este último caso, el manejo de flujos masivos de personas ha sido un inconveniente para distintas instituciones, debido a que los tiempos de espera pueden llegar a ser muy largos en algunos casos, causando disconformidad en los usuarios. Es por ello que las instituciones elaboran estrategias para brindarles una mejor calidad de servicio al optimizar los tiempos de espera.

De acuerdo a lo anterior, este proyecto de titulación tiene por objetivo responder a la necesidad planteada elaborando un sistema que sea capaz de reservar tickets de atención a través de una aplicación móvil y luego hacer el seguimiento vía Notificaciones Push del estado en cualquier instante de la fila de espera, informando el número de personas que restan por atender para así acudir a la oficina sólo en el momento preciso en que será llamado el cliente. En este sentido, el cliente cuenta con la oportunidad de aprovechar mejor su tiempo realizando otros trámites mientras espera su turno de atención. Todos estos dispositivos almacenarán y consumirán información de un servidor central. A su vez, la información entregada por los dispositivos será utilizada para realizar reportes que servirán para análisis posteriores por parte del centro de atención. En consecuencia, este proyecto crea una ventaja competitiva en el ámbito de la calidad de servicio, donde lo más probable es que exista un tiempo de espera considerable donde uno deba realizar un trámite.

## **1.1. Antecedentes**

Actualmente, los sistemas más recientes en cuanto a la atención de flujos de personas en el mundo, están compuestos habitualmente por:

- Pantallas indicadoras basadas en monitores o televisores LCD donde el cliente, en espera de su turno, las observa hasta que sea llamado su número.
- Dispensadores de tickets con monitores TouchScreen e impresoras térmicas de auto corte, donde el cliente realiza la petición del servicio.
- Servicio de cola virtual en el Banco de Venezuela.

El proceso comienza con la llegada del cliente al local. Inmediatamente accede a un dispensador para solicitar su ticket de atención para luego esperar hasta que observe y/o

escuche que ha llegado su turno.

Sistemas más antiguos se diferencian de los actuales en el aspecto tecnológico, ya que la idea central del proceso es la misma.

A pesar que esta solución permite organizar de una mejor manera el proceso de atención y con ello optimizar los tiempos de espera ya que el cliente al sacar su ticket debe permanecer en el local hasta ser atendido, pudiendo ocupar ese tiempo en realizar otras diligencias de igual o mayor importancia para así poder administrar de mejor manera su tiempo disponible.

En el servicio del Banco de Venezuela, “los clientes solicitan tickets de atención a través de mensajería de texto. Los SMS enviados al 2662 deben contener la combinación de caracteres TCK, acompañado de las letras V, E o P seguido del número de documento de identidad correspondiente (cédula o pasaporte), para a partir de allí recibir un número de referencia a ser indicado en las máquinas de entrada a las agencias bajo la opción Cola Virtual.”[Ins12]

En Brasil la Ley de Consumo protege a los clientes contra las extensas filas de espera en entidades bancarias. Ley nº 8.078/90, en el Artículo 14, Enunciado N.º 2.7 expresa claramente que la ley municipal impone un límite temporal de espera en la fila de una sucursal bancaria. Éste al no cumplirse, provoca daño moral al cliente que a su vez, en tal caso, tiene la posibilidad de una indemnización por un valor de R\$1.000 ( mil reales) a título de daños no patrimoniales[Pre90]. En el Anexo A se encuentra la traducción a español de un extracto de la Ley nº 8.078/90.

Dado lo anterior, se concluye que está presente la voluntad por cuidar al cliente. Se dispone de lugares confortables para que éste espere mientras es llamado para ser atendido o, como es el caso de Brasil, se protege al cliente imponiendo un límite de espera máximo que en caso de no cumplirse tiene como consecuencia una indemnización por parte del banco. Si bien lo anterior representa un avance, se pueden continuar mejorando estos sistemas que mejoran la calidad de vida del consumidor haciendo uso de las nuevas posibilidades que nos proporciona la tecnología hoy en día.

## **1.2. Motivación**

La motivación principal para este proyecto es que tenemos la oportunidad de mejorar los procesos de atención a clientes, específicamente en filas de espera en centros de atención cuyo proceso de atención sea similar, de tal modo que tengan la posibilidad de abandonar las dependencias y realizar otros trámites.

El desarrollo de este sistema permitirá el registro a través de una aplicación móvil y seguimiento vía Notificaciones enviadas al Smartphone de tickets de atención. Además, con la información almacenada en el servidor, se realizarán reportes que mostrarán estadísticas relacionadas con tiempos de atención de los diferentes servicios que presta el centro de atención involucrado.

No existen soluciones en nuestro país que resuelvan este problema. Existe una oportunidad real para diferenciar la calidad de atención que ofrece el centro al cual acude una persona para realizar un trámite. El trabajo contempla un gran énfasis en la computación móvil, pudiendo recibir notificaciones en un Smartphone en cualquier zona donde tenga servicio.

## **1.3. Impactos**

El desarrollo de tecnologías que apuntan al paradigma del Internet de las cosas es un área poco explorada y con un potencial enorme de utilidad para toda la población. Por ejemplo, controlar la iluminación de nuestro hogar, a través de un Smartphone, junto con encender y apagar aparatos eléctricos conectados a un enchufe inteligente [Kit].

Con el sistema a implementar, los clientes tienen la oportunidad de organizar mejor su tiempo y realizar otros trámites mientras esperan a ser llamados. Esto repercute directamente en un mejoramiento de la calidad de vida de los involucrados. Las personas podrán hacer un mejor uso de su tiempo, tema no menos importante en el contexto de una sociedad interconectada en la que vivimos.

## **1.4. Definición del Proyecto**

### **1.4.1. Objetivo general**

Desarrollar un sistema capaz de solicitar un número de atención a través de una aplicación para Smartphones con Sistema Operativo Android y hacer el seguimiento vía Notificaciones Push de los tickets de espera para personas que estén en alguna oficina de atención, con objeto de disminuir tiempos de espera.

### **1.4.2. Objetivos específicos**

1. Analizar las principales tecnologías Web y sistemas de notificación a dispositivos móviles, que permitan registrar y realizar el seguimiento de tickets de espera.
2. Obtener, analizar y especificar los requisitos del sistema, además de analizar y diseñar casos de uso correspondientes con los objetos de diseño necesarios, y generar mockups de la interfaz para el posterior desarrollo del registro vía: Plataforma Web y aplicación Android, junto con el seguimiento a través de Notificaciones Push de los tickets de espera para los usuarios del centro de atención donde se encuentren.
3. Implementar el prototipo funcional de acuerdo a lo modelado en los objetivos específicos anteriores.
4. Generar un Sistema Web de reportes, que permita gestionar los datos obtenidos en este proceso.

## **1.5. Organización del documento**

El contenido del documento se encuentra dividido en 7 capítulos organizados de la siguiente manera:

- El primer capítulo aborda la introducción del proyecto de título.
- El segundo capítulo, llamado “Marco teórico”, profundiza en los contenidos relevantes que son tratados a lo largo del documento, referentes a *Google Cloud Messaging* para Android, Notificación Push, Android, MySQL, *Web Service*, el Internet de las cosas, dispositivos móviles, proporcionando definiciones y llevando estas al contexto del proyecto. Finalmente, se realiza una revisión del estado del arte

pertinente a estos sistemas y contexto **de acuerdo al primer objetivo específico del proyecto.**

- El tercer capítulo, llamado “Solución propuesta”, define la problemática que plantea la gestión de tickets de atención, tomando en cuenta la información recabada en el objetivo uno y se propone una solución a la situación expuesta. Además, se realiza la selección de las herramientas de *software* a utilizar para la implementación de un prototipo funcional que abarque la problemática definida.
- El cuarto capítulo, llamado “Análisis y diseño”, aborda las tareas necesarias para modelar la solución propuesta dada la problemática definida, y especifica el uso de las tecnologías seleccionadas en el capítulo anterior, todo esto **de acuerdo al segundo objetivo específico del proyecto.**
- El quinto capítulo, llamado “Implementación”, expone la construcción del prototipo de software **de acuerdo con el cuarto objetivo específico**, según el proceso de análisis y diseño llevado a cabo en el tercer y cuarto capítulo.
- El sexto capítulo, llamado “Pruebas y ajustes”, define un método para realizar pruebas y ajustes necesarios del sistema desarrollado, y se presentan los resultados de la prueba previamente definida.
- Finalmente, el séptimo capítulo aborda las “Conclusiones” obtenidas del trabajo realizado, además de proyectar posibles mejoras para llevar a cabo en una posible continuación de este proyecto.

## 2. MARCO TEÓRICO

En este capítulo, se describen las tecnologías y conceptos asociados al proyecto, comenzando con la descripción del Sistema Operativo *Android*, posteriormente se da paso a explicar claramente el paradigma de el Internet de las Cosas. Finalmente, se realiza una revisión al estado del arte relativo a sistemas similares de gestión de filas de atención que existen en la actualidad.

### 2.1. Tecnologías Asociadas

#### 2.1.1. Android

*Android* es un sistema operativo basado en un *kernel* de *Linux*, su interfaz de usuario está basada en la manipulación directa por pantalla táctil, diseñado para teléfonos inteligentes y *tablets*. El sistema operativo utiliza el ingreso táctil para simular acciones del mundo real, como arrastrar, pinchar y presionar para manipular objetos en pantalla. Sin embargo, ha sido incorporado a otros dispositivos como: televisores, cámaras digitales, consolas de videojuegos, entre otros.

El sistema operativo fue desarrollado por la empresa *Android Inc.*, una firma comprada por *Google* el año 2005. *Android* se desarrolla de forma abierta, a diferencia de *iOS* o *Windows Phone*, y se puede acceder al código fuente y también a la lista de incidencias donde se pueden ver los problemas no resueltos y reportar nuevos.

Desde el año 2011 *Android* tiene la mayor cantidad de instalaciones en dispositivos móviles, y desde el 2013 estos dispositivos se venden más que todos sus competidores combinados [Bus14]. En julio del mismo año su tienda de aplicaciones, *Google Play*, alcanzo el millón de aplicaciones publicadas, y más de 50 mil millones de aplicaciones descargadas [Pho13]. Una encuesta de desarrolladores realizada entre abril y mayo del 2013 concluyó que el 71 % de los desarrolladores de aplicaciones móviles desarrolla para *Android* [Dev13]. En la Tabla 1 se pueden apreciar las versiones de *Android* a las que Google brinda soporte en la actualidad, su nombre clave, su fecha de lanzamiento y penetración de mercado en el total de dispositivos al 5 de Febrero del 2015.



Tabla 1. Versiones de Android soportadas por Google en la actualidad [And15].

Versión	API	Nombre Clave	Fecha de lanzamiento	Distribución
5.0	21	Lollipop	25 de Junio, 2014	1.6 %
4.4	19	KitKat	31 de Octubre, 2013	39.7 %
4.3	18	Jelly Bean	24 de Julio, 2013	6.3 %
4.2	17	Jelly Bean	13 de Noviembre, 2012	19.8 %
4.1	16	Jelly Bean	9 de Julio, 2012	18.4 %
4.0.3 - 4.0.4	15	Ice Cream Sandwich	16 de Diciembre, 2011	6.4 %
2.3.3 - 2.3.7	10	Gingerbread	9 de Febrero, 2011	7.4 %
2.2	8	Froyo	20 de Mayo, 2010	0.4 %

En cuanto al desarrollo de aplicaciones para *Android* habitualmente se implementan en el lenguaje de programación *Java*, utilizando alguno de los siguientes entornos de desarrollo: *Android Software Development Kit (Android SDK)* o *Android Software Development Kit (Android Studio)*. También se pueden usar otros lenguajes y herramientas de desarrollo como *Visual Basic* utilizando el *SDK Basic4Android*. Además se puede utilizar C#, .NET. Para estos últimos lenguajes se pueden usar los *SDK Visual Studio*, *MonoDevelop* o *Xamarin Studio*.

Android SDK es un conjunto de herramientas de desarrollo de aplicaciones para *Android*. Esta incluye un *debugger*, librerías, un emulador de dispositivo, documentación, código de ejemplo y tutoriales para apoyar el desarrollo, además de incluir en la actualidad el entorno de desarrollo integrado *Eclipse*. Estas herramientas se pueden utilizar en sistemas basados en *Microsoft Windows*, *Mac OSX* y *Linux*.

**2.1.2. Google Cloud Messaging para Android**

GCM (*Google Cloud Messaging*) es un servicio gratuito de mensajería, que permite el envío de notificaciones desde un servidor a una aplicación *Android*. Puede tratarse de un mensaje corto que indique un mensaje sencillo. Los mensajes tienen un tamaño de hasta 4 KB de datos [And14].

GCM reemplaza a la versión beta de C2DM, anterior servicio de mensajería nube –

dispositivo de *Android*. C2DM está obsoleto, y ya no está en funcionamiento para aplicaciones nuevas.

2.1.2.1. Arquitectura

GCM esta compuesto por un servidor de conexión proporcionado por Google (GCM Connection Servers), un servidor 3rd – Party App Server que interactúa con el servidor de conexión y una aplicación cliente (*Client App*) que se ejecuta en el dispositivo Android. La estructura de la arquitectura antes descrita se puede ver en la Figura 1.

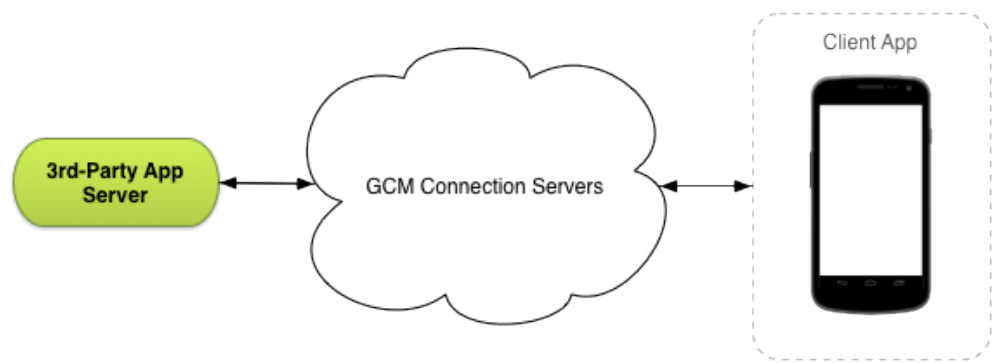


Figura 1. Arquitectura de GCM [And14].

En la Tabla2, se detalla cada uno de los componentes de la arquitectura de GCM mostrados en la Figura1.

Tabla 2. Componentes de GCM

Componentes	
Client App	La aplicación es ejecutada sobre un dispositivo <i>Android</i> . El dispositivo debe contar con <i>Android 2.2</i> y debe tener iniciada una sesión con su cuenta de <i>Google</i> .
3rd-Party Application Server	Es un servidor (es implementado por programadores que deseen utilizar el servicio de GCM) que envía datos a una aplicación <i>Android</i> en un dispositivo a través del servidor GCM.
GCM Connection Servers	Son servidores que provee <i>Google</i> , y se encargan de recibir los mensajes desde el servidor <i>3rd-Party Application</i> y enviarlos al dispositivo <i>Android</i> donde se encuentra la aplicación cliente ( <i>Client App</i> ).

#### **2.1.2.2. Enviar Mensajes**

A continuación se muestra la secuencia general de eventos que ocurren cuando nuestro servidor envía un mensaje:

1. Nuestro servidor (3rd-Party Application) envía el mensaje a los servidores GCM de Google.
2. Google guarda el mensaje en una cola de espera en caso de que el dispositivo Android esté offline.
3. Cuando el dispositivo Android está online, Google le envía el mensaje.
4. En el dispositivo Android, el sistema se encarga de que el mensaje sea recibido por la aplicación correcta utilizando los permisos correspondientes. Lo anterior despierta a la aplicación Android implicada. La aplicación no necesita estar ejecutándose para recibir el mensaje.
5. La aplicación Android procesa el mensaje.

#### **2.1.2.3. Recibir Mensajes**

Esta es la secuencia de eventos que ocurren cuando una aplicación Android instalada en un dispositivo móvil recibe un mensaje:

1. El sistema recibe el mensaje entrante y extrae una API key junto con los datos del mensaje.
2. El sistema pasa la API key y los datos a la aplicación correspondiente.
3. La aplicación extrae los datos asociados a la API key y procesa la información contenida.

#### **2.1.3. Notificaciones Push**

La tecnología *Push*, es un estilo de comunicación sobre internet donde la petición de una transacción comienza en el servidor.

Los servicios *Push*, generalmente, están basados en información a medida. Es decir, funcionan bajo un modelo publicador – suscriptor. Un cliente se suscribe a un medio

de comunicación y, cuando haya nuevo contenido disponible el servidor deberá enviar la información al usuario.

La mensajería Push presenta dos ventajas:

1. La notificación es instantánea, los mensajes se reciben de inmediato.
2. Las notificaciones *Push*, no necesitan que la aplicación receptora esté ejecutándose en el dispositivo y realizando consultas al servidor consultando por nueva información. Esto permite que la aplicación se esté ejecutando en segundo plano y se active cuando reciba un mensaje.

*BlackBerry* fue la primera plataforma móvil que integró la tecnología *Push* para el servicio de correo electrónico en sus dispositivos. En el año 2009, *Apple* lanzó su propia infraestructura de notificaciones mediante tecnología *Push*, a través de APNs (*Apple Push Notification service*) . La última plataforma en incorporar éste servicio fue *Android* a través de C2DM ( *Cloud to Device Messaging*) de *Google*. Dicha tecnología se encarga de la gestión de cola y envío de mensajes a la aplicación cliente. Hace poco tiempo *Google* actualizó su plataforma de notificaciones. Ahora se llama GCM (*Google Cloud Messaging*) que sustituye a C2DM. Los requisitos para los dispositivos móviles que quieran utilizar la infraestructura son disponer una versión de *Android* 2.2 o superior y tener instalado *Google Play*. Estos mensajes se muestran en nuestro dispositivo por ejemplo cuando recibimos un mensaje SMS o tenemos una llamada perdida. Constan de un **icono** y un **texto** que se muestra en la barra de estado de *Android* junto con un **mensaje** descriptivo y una marca que contiene la **hora** en que llegó la notificación. A modo de ejemplo, cuando tenemos una notificación de alguna aplicación en nuestra barra de estado, por un lado vemos el icono junto a un texto principal y más abajo un segundo texto con un mensaje de carácter descriptivo; lo anterior, se puede apreciar en la Figura2.

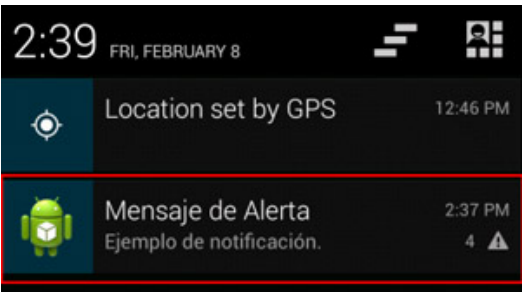


Figura 2. Ejemplo de Notificación Push en Android. [Sgo11g].

### 2.1.4. Web Service

Un servicio web ( *Web Service* en Inglés ) es una tecnología que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Éstas aplicaciones pueden estar desarrolladas en diferentes lenguajes, o pertenecer a plataformas distintas. Para poder intercambiar información a través de redes de computadores se utilizan los servicios web. Éstos servicios aportan interoperabilidad entre aplicaciones, es independiente de las plataformas en donde estén instaladas. Permiten que software de diferentes compañías, ubicadas en distintas partes del mundo, puedan ser combinados y proveer servicios integrados.

A modo de ejemplo, la Figura 3 corresponde a un esquema donde las flechas rojas indican el flujo de datos ante una **solicitud** de una aplicación *Android* a una base de datos MySQL, realizada a través de un web service escrito en *PHP* y el respectivo flujo de datos de **respuesta**, flechas verdes, enviados desde la base de datos hacia la aplicación móvil utilizando el formato de intercambio de datos JSON.

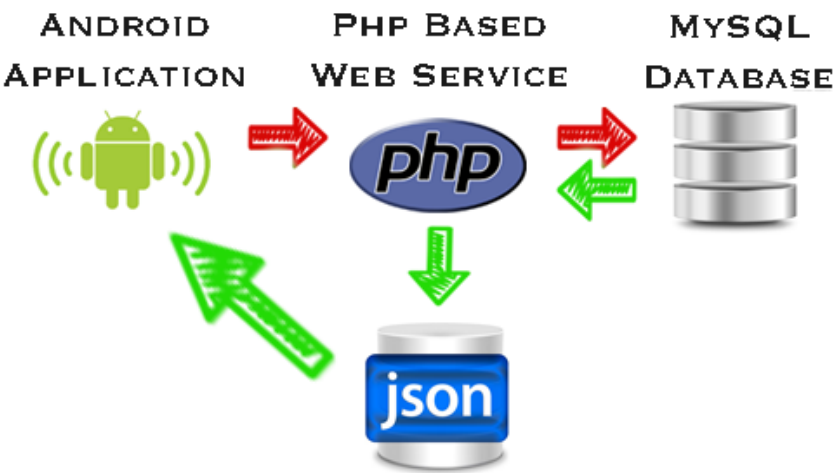


Figura 3. Flujo de datos entre una aplicación *Android*, *Web Service PHP* y MySQL. [Myb13].

### 2.1.5. MySQL

Es un sistema de gestión de base de datos relacional. MySQL<sup>1</sup> es utilizado en aplicaciones web y de escritorio. Sin embargo se puede conectar con aplicaciones móviles mediante el uso de *Web Service*. Es uno de los sistemas de bases de datos de código abierto más

---

<sup>1</sup><http://www.mysql.com/>

populares, existe bastante documentación sobre cómo conectar MySQL con: servidores web y servicios web escritos en diversos lenguajes de programación, además de librerías que soportan la comunicación con aplicaciones para varias: plataformas y lenguajes de programación.

## 2.2. Teoría de Colas

Hoy en día estamos rodeados de sistemas de colas o filas de espera, cuando estamos en un taco vehicular, un semáforo mal regulado, un peaje, esperamos en el teléfono a ser atendidos por un operador, entre otros. En el sistema a desarrollar se quiere proporcionar información al cliente sobre el estado de avance de la fila donde éste solicitó un número de atención. Como por ejemplo, advertir un tiempo de atención estimado por cada cliente atendido. Para ello, es necesario considerar esta teoría en el prototipo a desarrollar.

La teoría de colas se dedica al estudio matemático de colas o líneas de espera dentro de un sistema. Estudia elementos como el tiempo de espera promedio o la capacidad de trabajo que tiene un sistema sin que colapse. Esta teoría es utilizada en una variedad de áreas como: telecomunicaciones, comercio, cadenas productivas, etc..

Un sistema de colas puede ser descrito como **clientes** que llegan a un centro en busca de un **servicio**, deben esperar si éste no es entregado de inmediato, y abandonan luego de ser atendidos. En algunos casos los clientes se retiran del sistema debido a un tiempo de espera excesivo. Son cuatro los componentes que definen la estructura básica de un sistema de colas[Sch04]:

1. Modelos de llegada: fijos o aleatorias según un patrón de comportamiento.
2. Modelos de servicio: fijo o definido por un tipo de distribución.
3. Disciplina de la cola: los clientes son atendidos dependiendo el orden de llegada al sistema:
  - a) FCFS (*First Coming First Served*:) El primer cliente en llegar es el primero en ser atendido.
  - b) LCFS (*Last Coming First Served*:) El último cliente que llega es el primero en ser atendido.

c) PS (*Processor Sharing*;) Procesamiento compartido.

4. Notación de Kendall (A/B/c/k/m) donde:

- a) A distribución de llegadas.
- b) B distribución de tiempo de servicio.
- c) c número de canales de servicios.
- d) k capacidad de la cola.
- e) m tamaño de la población.

Por lo anterior se puede deducir que un sistema de colas es altamente complejo. Sin embargo, para este proyecto se considerará un modelo simple en el que habrá una fila de espera para cada canal de atención (servicios). Finalmente, se calculará el tiempo medio de espera para cada servicio. Éste, será enviado en las notificaciones a cada dispositivo cada vez que la fila de espera avance.

### 2.3. Internet de las cosas

El término de Internet de las cosas no es conocido por las personas que utilizan tecnología a diario: desde computadores, termostatos y teléfonos inteligentes, hasta una maleta inteligente que está dotada de GPS, y sensores que se comunican vía *Bluetooth* con el celular y la red [Inf14].

Para *Cisco Internet Business Solutions Group*, el gigante mundial en infraestructura de red, lo define como “simplemente es el punto en el tiempo donde comenzaron a haber más ‘cosas u objetos’ conectadas a Internet que personas [Cis11].”

También se puede entender este concepto pensando en la capacidad que tiene un objeto o cosa inteligente que puede comunicarse con las personas. Este es un escenario donde animales, personas y objetos están todos conectados y provistos de un identificador único que permite que sus datos sean almacenados en un sistema. De esa forma, podemos acceder a los datos asociados a esa entidad e interactuar con ellos. Como consecuencia, existe la posibilidad de transferir datos sobre éstos a través de la red sin necesidad de interacción: persona - persona o persona - ordenador. Lo anterior es posible gracias a la

evolución de las tecnologías inalámbricas e Internet.

Lo descrito anteriormente se puede ver en la Figura 2, donde se aprecia claramente la diversidad de objetos que son considerados en este paradigma.



Figura 4. Internet de las cosas [The14].

**2.3.1. Big Data**

Los computadores y los dispositivos móviles han dejado de ser los únicos objetos capaces de comunicarse con Internet. Lo que antes era algo futurista, es ahora una realidad. Automóviles, electrodomésticos y otros objetos nunca imaginados ahora también se pueden conectar a la red. Y con su masificación, nuevos productos y servicios irán surgiendo conforme a las posibilidades tecnológicas se perfeccionen.

Esta unión entre el mundo de las cosas y el mundo virtual representará un incremento vertiginoso del *Big Data*. Es decir, la tecnología de procesamiento de millones de datos en tiempo real, provenientes de diferentes ámbitos como: negocios, gobierno, servicios financieros, medicina, ciencia e ingeniería.

Las soluciones tecnológicas para analizar esa cantidad abismal de datos, es clave para el desarrollo de las empresas dentro de la nueva situación de mercado de hoy en día. En este contexto, la movilidad y la nube tendrán un rol fundamental: mientras los smartphones u objetos de los consumidores, conectados a la red, emiten volúmenes importantes de información, la nube será el lugar donde se alojarán. La ventaja que brindan las soluciones analíticas radica en que las empresas podrán tomar decisiones más acertadas y ofrecer a sus clientes lo que ellos necesitan.



El año 2012, investigadores de IBM se unieron con la ciudad de Lyon en Francia para crear un sistema que ayude a los agentes de tráfico a reducirlo en las calles de la ciudad. El sistema *Decision Support System Optimizer*, utiliza informes en tiempo real sobre el tráfico para detectar posibles tacos. De tal modo, si un agente detecta que puede haber un atasco en algún punto, puede cambiar las señales de tráfico para mantener las carreteras con un flujo constante [IBM12].

En el contexto del proyecto, los clientes que asisten a los centros de atención, donde existe una atención masiva de clientes en los diversos servicios que proporcionan, entregarán información valiosa a éstos, a través de sus dispositivos móviles, dándoles la posibilidad de tomar decisiones acertadas y oportunas para reducir los tiempos de atención en las filas de espera de los servicios más requeridos.

## **2.4. Estado del Arte**

En este apartado, se revisarán sistemas similares al que se pretende desarrollar en el proyecto, que signifiquen un aporte parcial para su posterior implementación. Analizar estos sistemas significan un apoyo para considerar las funciones más comunes y detectar características relevantes que se pueden incluir en el proyecto.

### **2.4.1. Banco de Venezuela, Cola Virtual**

Es un servicio implementado por el Banco de Venezuela, que permite a clientes y no clientes enviar un mensaje de texto desde su celular, mediante un sistema automatizado de administración de cola, solicitando un número de ticket, mucho antes de llegar a la oficina bancaria, para ser atendido por ventanilla. Va dirigido a personas naturales, clientes o no clientes de dicho banco.

Su uso no implica cobros por parte del Banco, pero se debe pagar por el costo del mensaje de solicitud de ticket al operador de telefonía. Es compatible con cualquier celular que permita enviar y recibir mensajes de texto (SMS), independiente del operador telefónico.

El proceso de petición de ticket es bastante engorroso. Hay dos formas de hacerlo, la

primera es para celulares normales y la segunda para teléfonos inteligentes.

#### **2.4.2. WAVETEC, Sistema de Administración de Filas**

La empresa en cuestión, tiene una solución para las personas que tienen problemas como: el exceso de espera desorganizado, el tiempo de espera de los clientes son cada vez mayores, los empleados tienen momentos improductivos, sus clientes se molestan por un servicio que no es el que esperaban, es incapaz de monitorear el rendimiento de una sucursal desde diferentes lugares.

Sus soluciones son ideales para Bancos, Centros de Servicio en compañías de Telecomunicaciones, Instituciones Gubernamentales, Hospitales, Empresas del Retail, Aerolíneas, Embajadas u organizaciones con un alto flujo de clientes. Wavetec ayuda a las empresas mas importantes de la industria para mejorar la experiencia del cliente alrededor del mundo. La solución integra diferentes tecnologías para seguir el estado de la fila: notificación vía SMS que el turno será llamado dentro de poco, permite a los clientes salir de las oficinas a otro lugar, visitar otros almacenes, etc. También se pueden solicitar turnos vía Web y SMS.

##### **2.4.2.1. Reportes en tiempo real**

El sistemas de Administración de Filas está equipado con un potente software de reportes que permitirá a las personas que toman las decisiones en las organizaciones actuar con valiosa información. Esta herramienta ha sido desarrollada para gerentes y ejecutivos para administrar el flujo de trabajo de sus áreas de servicio al cliente en cada sucursal a través de información de inteligencia de negocios, por ejemplo, tiempos de espera promedio, numero exacto de clientes en espera, tiempos de ausencia de los asesores y muchos mas indicadores de desempeño.

##### **2.4.2.2. Reportes de sucursales**

Se pueden resumir por 5 periodos de tiempo (por hora, día, semana, mes, año).

1. **Reportes Resumidos:** Muestra el rendimiento en servicio de una sucursal en un momento determinado.
2. **Reportes por Categoría:** Entrega información del flujo de clientes en cada categoría de servicio.
3. **Reportes por Operador:** Entrega información sobre el rendimiento de un operador en la sucursal.
4. **Reportes de Alertas:** Este informe compara el rendimiento de la sucursal contra los indicadores de desempeño (KPI) de tiempo de espera, tiempo de duración de la atención y la capacidad de asientos en la sala de espera.

#### **2.4.3. Q-sige, Sistema Inteligente de Gestión de Esperas**

El Sistema Inteligente de Gestión de Esperas, está compuesto por diversos programas que se comunican a través de la red. Cuenta con un módulo de estadísticas (SIGEST) y un módulo de **Cita Prévia (web o Telefónica)** mediante el cual, el cliente puede reservar una hora. Permite optimizar la calidad de atención al público. Su ventaja principal es, además de reducir los tiempos de espera, transformar las incómodas filas en zonas de espera confortables para los clientes. En estas cómodas zonas de espera, se les informa a las personas cuándo y dónde se les atenderá. [Idm14].

Otra funcionalidad del sistema SIGE es proveer de información a los responsables de la gestión de calidad de la atención al público, aportando con información que permite conocer lo que está ocurriendo en un instante de tiempo y, reaccionar oportunamente para analizar y optimizar sus recursos.

Este sistema cuenta con dos versiones. Una versión reducida de Q-sige, diseñado para negocios pequeños con un espacio reducido. La versión completa se llama Q-sige Synergy, está orientada a empresas con un gran número de oficinas y servicios.

### **3. SOLUCIÓN PROPUESTA**

En este capítulo, se expone la problemática que aborda este proyecto de tesis. Además, se muestra la solución que se llevará a cabo y luego las herramientas que posibilitarán el desarrollo de lo propuesto anteriormente, con el objetivo de gestionar y reducir los tiempos de espera en diferentes centros de atención altamente concurridos.

#### **3.1. Problemática**

Es recurrente que al momento de realizar un trámite o compras en un centro de atención: bancos, clínicas, sección de carnes en un supermercado, etc., en horas punta exista una probabilidad alta en que el tiempo de espera sea considerable. Este tiempo excesivo, es un hecho generador de desgaste físico y emocional ocasionando daño moral en las personas. Es por ello que las instituciones elaboran estrategias para brindarles una mejor calidad de servicio al optimizar los tiempos de espera. Sin embargo, estos esfuerzos continúan requiriendo más incorporación de tecnología con tal de brindar mayores avances en esta temática.

#### **3.2. Presentación de la solución**

Para resolver la problemática antes presentada, se propone una solución de gestión de turnos de atención, que contempla su seguimiento en tiempo real. Ésta, consta de tres partes:

1. Una aplicación móvil que permite obtener los turnos de atención.
2. Almacenamiento de datos.
3. Un módulo web que permite gestionar los turnos de atención y generar reportes utilizando los datos almacenados.

La aplicación móvil, sirve como herramienta para obtener un turno de atención y recibir notificaciones con información del estado de avance de la fila que corresponde al servicio que seleccionó el usuario. Además, ésta envía datos relevantes del dispositivo móvil para ser almacenados en un servidor central.

En el párrafo anterior se menciona el almacenamiento de datos en un servidor central. Para hacer efectivo esto, el servidor cuenta con un sistema de gestión de bases de datos,

y una serie de aplicaciones que soportan la comunicación entre la aplicación móvil, dicho servidor y el módulo web.

El módulo web mencionado es un sistema que permite gestionar los turnos, coordinar el envío de las notificaciones a los *smartphones* para que los clientes se acerquen al módulo de atención correspondiente y generar reportes que muestran estadísticas sobre los diferentes servicios que brinda el centro de atención en cuestión.

### 3.3. Descripción de la metodología

En general el desarrollo del proyecto se basará en la búsqueda en Internet y el continuo apoyo del patrocinante, mostrando avances periódicos e incrementales. El desarrollo del proyecto contempla un marco de trabajo ceñido al ciclo de vida iterativo e incremental, debido a que el contacto con el profesor patrocinante será constante para validar los avances presentados. Además se realizarán varias pruebas o prototipos iniciales, para comprobar que está correcta la **comunicación/funcionamiento** entre las diferentes piezas de software que integran el sistema y así evitar posibles complicaciones que signifiquen excesivas horas de dedicación para resolver errores.

**Objetivo Específico 1:** Analizar las principales tecnologías web y sistemas de notificaciones que permitan registrar y realizar el seguimiento de tickets de espera.

Se comenzará investigando en Internet lo referente a las tecnologías de notificación disponibles para *Smartphones* y, además se hará una revisión de proyectos similares que estén en funcionamiento.

**Objetivo Específico 2:** Obtener, analizar y especificar los requisitos del sistema, además de analizar y diseñar casos de uso correspondientes con los objetos de diseño necesarios, y generar mockups de la interfaz para el posterior desarrollo del registro vía web y/o aplicación *Android* y seguimiento a través de Notificaciones *Push* de los *tickets* de espera para una oficina de atención de clientes.

Primero se deben especificar los requerimientos del sistema a implementar. Se incluirán artefactos UML, interfaz del *software* y un modelo de datos. Además, se debe modelar la

arquitectura que sostendrá este prototipo. Es decir, el *hardware* y *software* que usará el sistema.

**Objetivo Específico 3:** Implementar el prototipo funcional de acuerdo a lo modelado en los objetivos específicos anteriores.

Se deberá implementar el prototipo en base a todas las normas y especificaciones descritas anteriormente, con la construcción de una interfaz de usuario amigable, soportado por una base de datos disponible en el servidor del sistema.

**Objetivo Específico 4:** Diseñar e implementar un Sistema web de reportes, que permita gestionar los datos obtenidos en este proceso.

Primero se debe contar con la especificación de requerimientos para este sistema web de reportes e incluir artefactos UML. Después se desarrollará el sistema Web de reportes, a partir de la información obtenida y almacenada del proceso de: inscripción, notificación y atención. Esto permitirá gestionar los procesos de atención.

### **3.4. Descripción de Tecnologías a utilizar**

En esta sección, se retomará el tema tratado en el punto 2.1, con la diferencia que aquí se justificará el por qué de la elección de la tecnología escogida entre todas las disponibles.

#### **3.4.1. Entorno de desarrollo**

Un entorno de desarrollo integrado (IDE) permite a un programador escribir código fuente, compilar, interpretar o ambos en algunos casos. Para escribir la aplicación móvil de este proyecto, se utilizó el IDE Android Studio 1.0<sup>2</sup>. Básicamente fué elegido sobre el tradicional Eclipse pues todo apunta a que será el entorno de desarrollo recomendado por el equipo de *Android*.

---

<sup>2</sup><http://developer.android.com/tools/studio/index.html>

### 3.4.2. Intercambio de datos

Se usará el modelo cliente-servidor para el intercambio de datos entre la aplicación móvil, el servidor central y el almacenamiento de información desde el módulo web. Para hacer efectiva esta comunicación, se utilizará el servidor web HTTP Apache, que se encargará de recibir y responder las peticiones de: la aplicación móvil y módulo web.

Para manipular las peticiones, se usará un servicio web escrito en el lenguaje de programación PHP, que se ocupará de distinguir las solicitudes que llegarán y responder pertinentemente a ellas. Como por ejemplo, almacenar información de un dispositivo nuevo, u obtener la última notificación recibida.

La aplicación móvil necesita de un medio para comunicarse con el servicio web alojado en el servidor central. Para ello, ese medio corresponde a un formato ligero de intercambio de datos que sea sencillo de leer y escribir para humanos. Por lo tanto, se optó por utilizar la Notación de Objetos JavaScript JSON, debido a su simplicidad en relación a XML. JSON básicamente describe los datos con una sintaxis dedicada que se usa para identificar y gestionar los datos.

JSON consta de dos estructuras:

1. Una colección de pares nombre/valor. En la mayoría de los lenguajes se conoce como un objeto, registro o *struct*<sup>3</sup>, tabla *hash*.
2. Una lista ordenada de valores. Esto en la mayoría de los lenguajes se conoce como arreglos, listas.

A modo explicativo y con el propósito de aclarar la estructura de JSON, en las figuras a siguientes se entrega un detalle de los dos puntos antes descritos.

Un Objeto es un conjunto desordenado de pares nombre/valor. Comienza con { (llave de apertura) y termina con } (llave de cierre). El nombre es seguido por : (dos puntos) y los pares nombre/valor están separados por , (coma). Lo anterior, se aprecia en la máquina de estado finito en la Figura 5.

---

<sup>3</sup><http://c.conclase.net/curso/?cap=011>

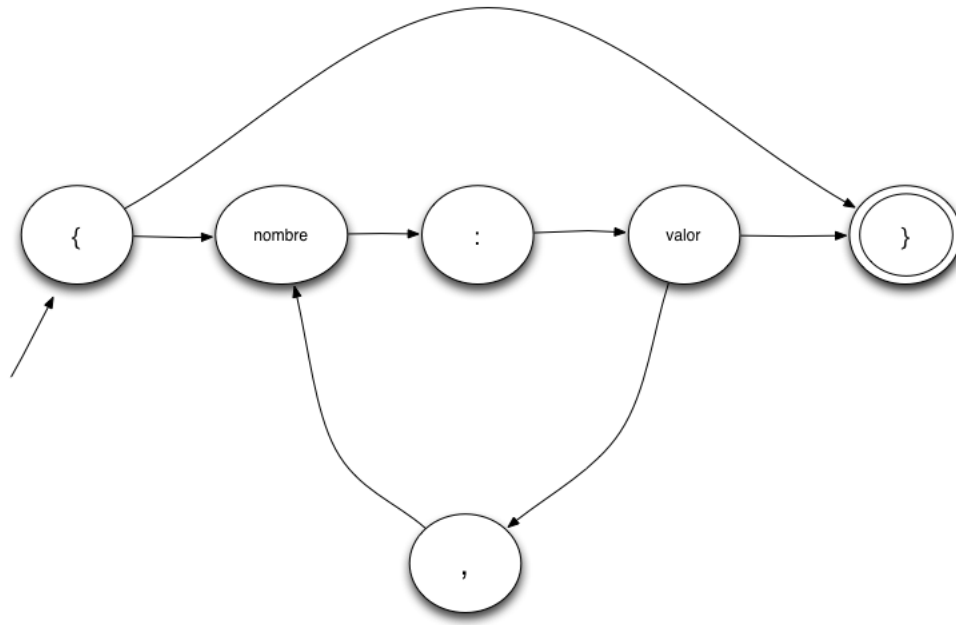


Figura 5. **Objeto:** conjunto desordenado de pares nombre/valor.

Un arreglo es una colección de valores, que comienza con [(corchete izquierdo) y termina con ] (corchete derecho). Los valores son separados por , (coma). Lo anterior, se aprecia en la máquina de estado finito en la Figura 6.

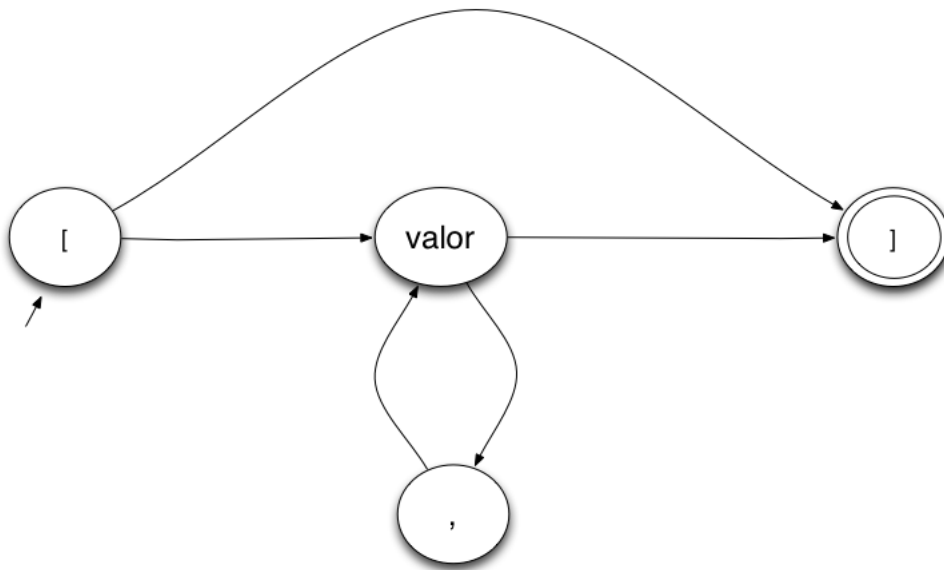


Figura 6. **Arreglo:** colección de valores separados por ',' (coma).

Un valor puede ser una cadena de caracteres, o número, u objeto, o arreglo, o true, o false, o null. Lo anterior, se aprecia en la máquina de estado finito en la Figura 7.



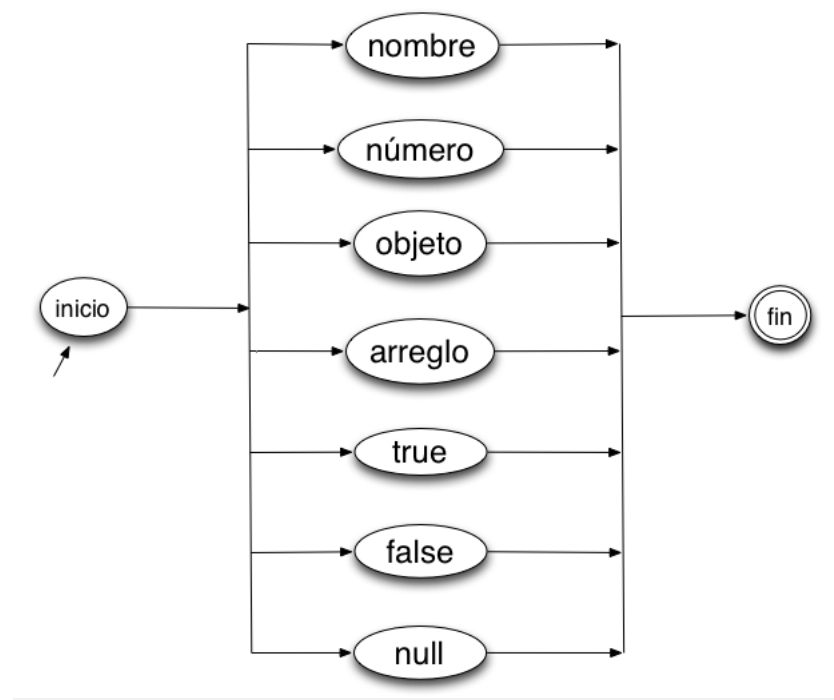


Figura 7. **Valor:** cadena de caracteres, o número, u objeto, o arreglo, o true, o false, o null.

3.4.3. Base de datos

Para almacenar los datos proporcionados por la aplicación móvil y el módulo web, se utilizará el sistema de gestión de bases de datos relacionales MySQL. En este apartado no se ahondará sobre este sistema pues fue descrito en el capítulo anterior, en el punto 2.1.5.

3.4.4. Librerías utilizadas

La única librería incluida en el proyecto y que hace posible la comunicación entre la aplicación móvil y GCM: enviar y recibir notificaciones es la librería de GoogleCloudMessaging. Ésta debe ser incluida en el proyecto en *Android Studio* para utilizar sus clases desde nuestra aplicación.

3.4.5. Patrón MVC

El acrónimo MVC corresponde a: **Modelo - Vista - Controlador**. Éste es un patrón de arquitectura de *software* donde en una aplicación separa los datos y la lógica del negocio de la interfaz de usuario y el módulo que se encarga de gestionar los eventos y comunicaciones. Por lo anterior, MVC propone que se construyan tres componentes diferentes: el modelo, la vista y el controlador y que al final se relacionarán para tener

como resultado una aplicación, en el caso del proyecto una aplicación móvil.

Para entender la Figura 8, se explicará brevemente de qué se trata cada uno de los componentes de este modelo.

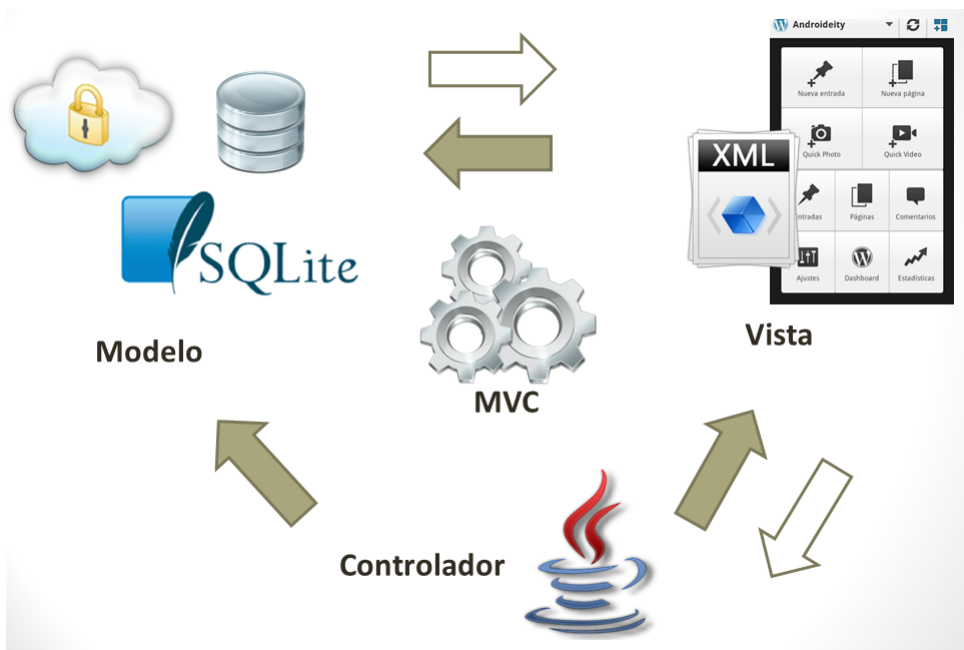


Figura 8. Patrón MVC.[And12]

1. **Modelo:** Básicamente el modelo corresponde a qué base de datos o servicios web se utilizará para almacenar la información. El modelo a elegir depende directamente de las necesidades de información que tenga la aplicación que se desarrollará.
2. **Vista:** La vista no es mas que la interfaz con la que va a interactuar el usuario. En *Android*, las interfaces se construyen en XML, y es a través de éstas que se presentan los datos a los usuarios.
3. **Controlador:** Son las clases que contienen el código y permiten darle vida a las interfaces construidas permitiendo desplegar y consumir información de/para el usuario. En un proyecto *Android* los controladores corresponden a las clases programadas en lenguaje JAVA y son el núcleo de la aplicación.

### 3.5. Arquitectura del sistema

En los puntos anteriores se ha descrito lo concerniente a cómo y qué tecnologías se utilizarán para dar solución al problema expuesto. Por último, para dar sentido a lo antes planteado en los tópicos anteriores, es factible definir una arquitectura para el sistema.

Ver Figura 9. La arquitectura planteada, se compone de un servidor central provisto de diversos servicios, *smartphones* que se comunicarán con el sistema, una base de datos para almacenar la información, un monitor que mostrará el turno que está siendo atendido, el servicio de GCM por el cual circulan las notificaciones y un módulo web de reportes.

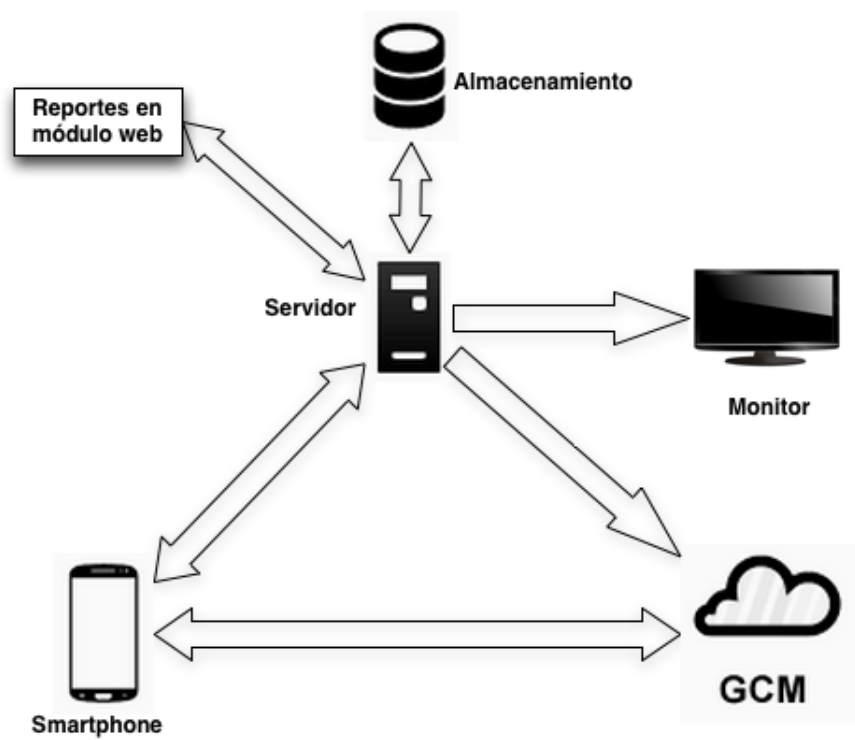


Figura 9. Arquitectura del sistema.

## 4. ANÁLISIS Y DISEÑO DEL PROTOTIPO DEL SISTEMA

En este capítulo se explican y detallan las etapas que permiten modelar la solución propuesta utilizando artefactos de *software*, que permiten documentar el sistema bajo el marco de trabajo de una metodología de desarrollo de *software*.

### 4.1. Metodología

Una metodología de desarrollo de *software* puede ser definida como un marco de trabajo que se utiliza para estructurar, planificar y controlar el proceso correspondiente al desarrollo de un sistema de información. Para una administración y gestión sistemática de todo proyecto de *software*, y llevarlo a cabo con altas posibilidades de éxito, es necesario guiarse por una metodología, permitiendo así dividir un proyecto en módulos o etapas. De esta forma estructurada y organizada es posible crear, desarrollar y mantener un sistema desde que surge la necesidad del producto hasta que se cumple el objetivo por el cual fue desarrollado.

En este proyecto se utilizará el ciclo de vida incremental para el desarrollo del *software*, ya que este modelo se acomoda bastante a la implementación de un prototipo, que es el caso de este proyecto de tesis. Además, no es necesario disponer de los requerimientos de todas las funcionalidades en el comienzo del mismo. El proceso consiste en ir construyendo módulos o iteraciones que van cumpliendo con las diferentes funciones del sistema, aumentando gradualmente las capacidades de éste. Las ventajas de este ciclo de vida son:

1. Desarrollar las funcionalidades por etapas permite detectar si los requerimientos planteados para los niveles siguientes son correctos.
2. En caso de enfrentarse ante un error grave solo se desecha la última iteración.
3. Construir por etapas, partes pequeñas del sistema, representa menos riesgo que construir un sistema grande.

### 4.2. Plan de trabajo

En la Figura 10 se ilustra el plan de trabajo para llevar a cabo la implementación del sistema:

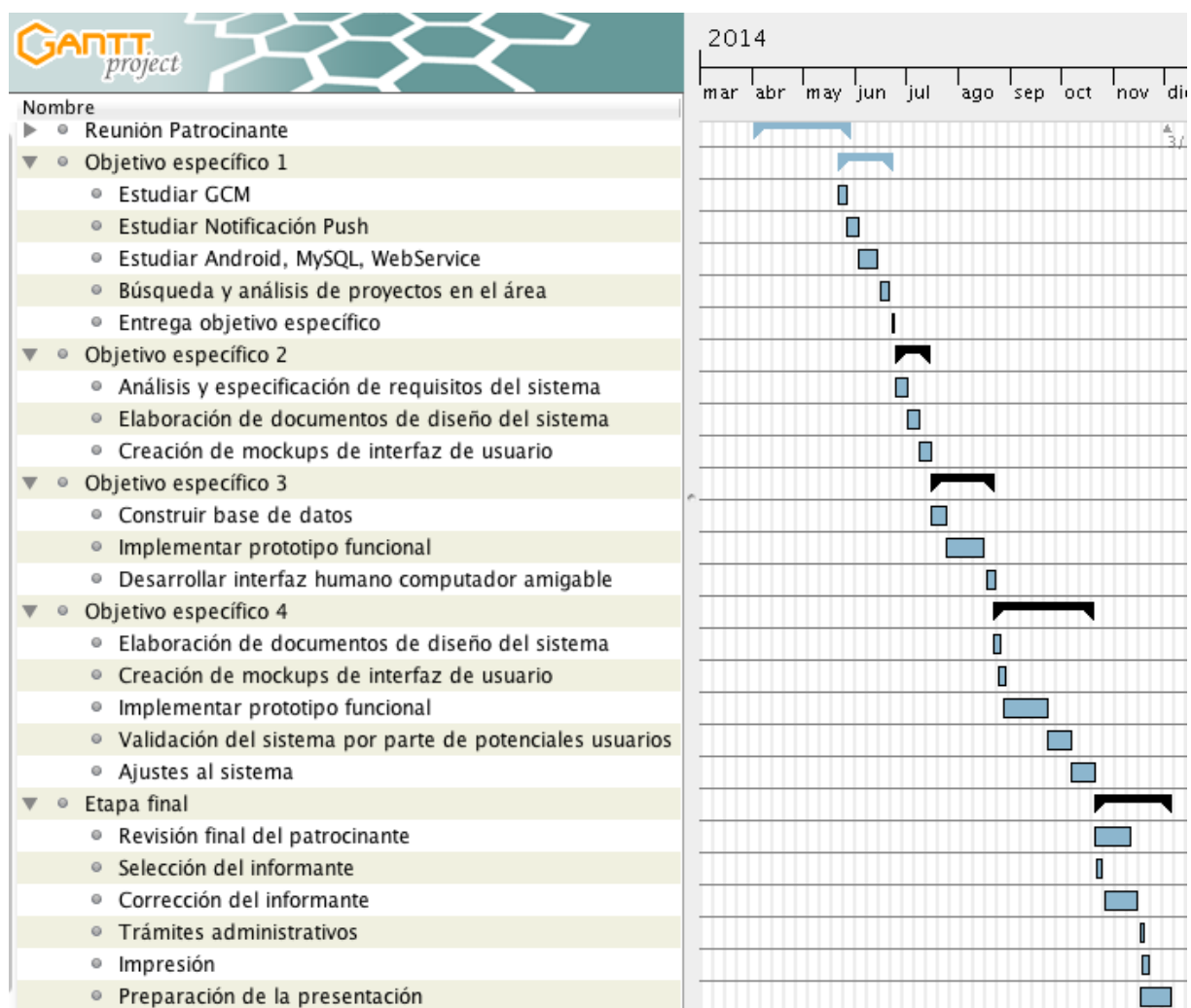


Figura 10. Carta Gantt que exhibe las etapas de desarrollo del sistema.

#### 4.2.1. Estrategia de implementación

A través del proceso de implementación y desarrollo del software, se tuvo que tomar en cuenta lo siguiente:

1. Es importante reunirse con el profesor patrocinante al comienzo, durante y al término de una iteración, para así aclarar y resolver requerimientos específicos del sistema que puedan surgir.
2. Durante el desarrollo del sistema, se consideran presentaciones de avance de las iteraciones de tal forma que se disminuyan errores y/o riesgos en la entrega final.
3. El producto final se dará terminado cuando se hayan cumplido todos los requisitos del sistema.

## 4.3. Análisis

### 4.3.1. Descripción de los actores del sistema

**Administrador:** es un individuo que se ocupa de configurar el sistema, antes de la puesta en marcha, y tiene acceso a todas las funcionalidades del mismo. Es decir, es un funcionario del centro de atención donde sea instalado el sistema de gestión de tickets.

**Ejecutivo:** es un funcionario del centro de atención que utiliza el sistema para realizar las llamadas a los clientes cuando sea oportuno. Tiene acceso limitado a la aplicación web.

**Cliente:** corresponde a una persona que interactúa como cliente con el sistema a través de la aplicación móvil desde su *smartphone* en un centro de atención.

4.3.2. Casos de Uso

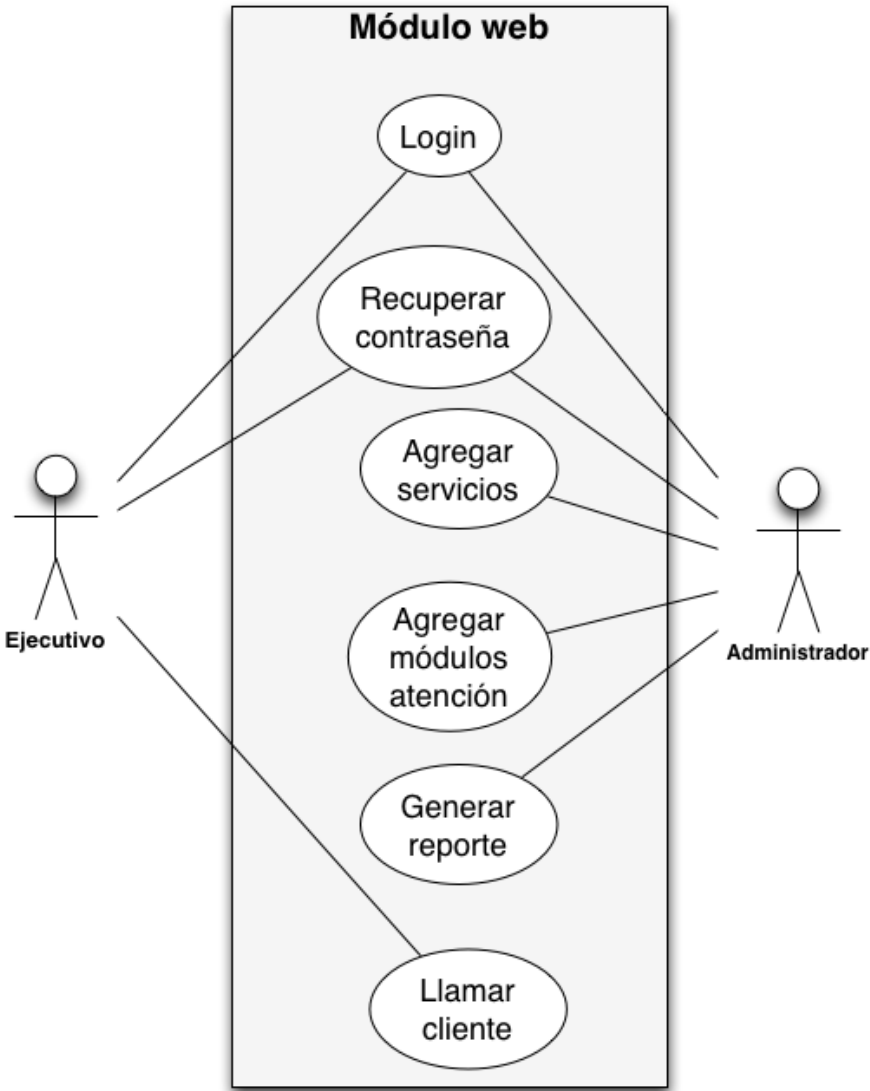


Figura 11. Diagrama de Casos de Uso Módulo web.

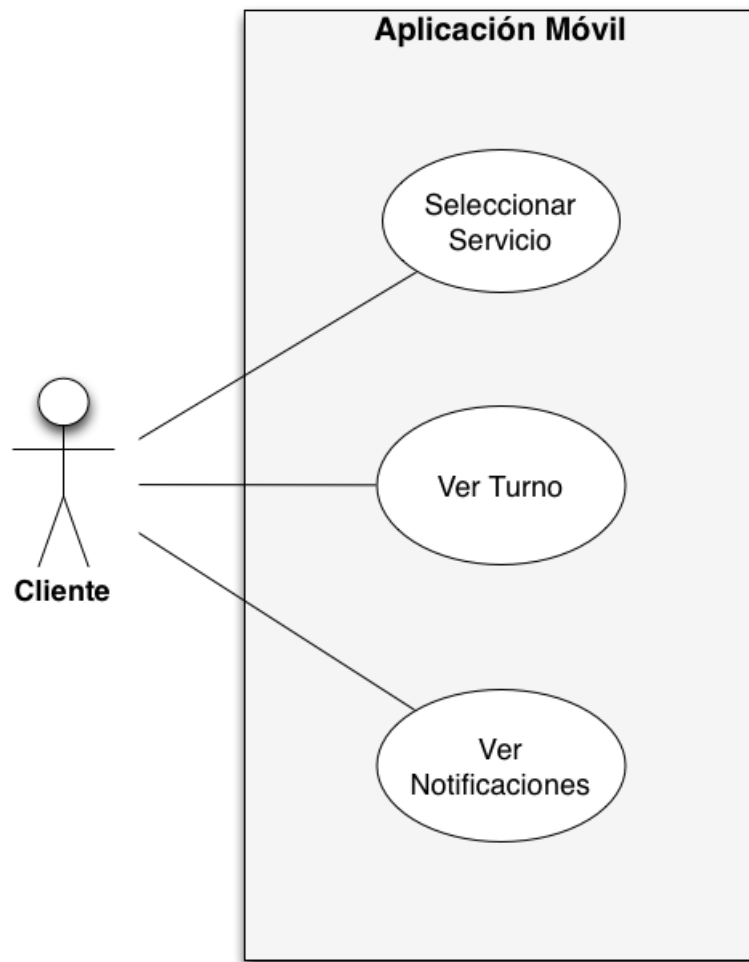


Figura 12. Diagrama de Casos de Uso Aplicación Móvil.

#### 4.4. Especificación de requerimientos

De acuerdo a la problemática y objetivos descritos se tomaron y obtuvieron los requisitos para poder desarrollar el Sistema de Gestión de tickets de espera.

##### 4.4.1. Requisitos Funcionales

Cuando se plantea una solución a una problemática, nacen ideas de cómo será y qué características tendrá el sistema que se desarrollará. Éstas características requeridas del *software* tienen relación con una capacidad de acción que tendrá el nuevo sistema, es decir, una funcionalidad.

A continuación en la tabla 3 se presentan los requisitos funcionales extraídos de los diagramas UML y de las capacidades que se desea que tenga el sistema a implementar.



Tabla 3. Requisitos Funcionales.

Ref. #	Función
FRQ-001	Iniciar sesión en aplicación web.
FRQ-002	Cargar y mostrar servicios.
FRQ-003	Cargar y mostrar módulos de atención.
FRQ-004	Enviar notificaciones desde aplicación web a <i>smartphone</i> .
FRQ-005	Generar reportes.
FRQ-006	Mostrar últimos turnos en atención.
FRQ-007	Monitor de atenciones.
FRQ-008	Mostrar servicios.
FRQ-009	Seleccionar un servicio.
FRQ-010	Ver turno de atención.
FRQ-011	Ver notificaciones.
FRQ-012	Mostrar notificaciones nuevas automáticamente.
FRQ-013	Mostrar información de la aplicación.

4.4.2. Requisitos No Funcionales

Los requisitos no funcionales, obedecen a todas las exigencias cualitativas que se imponen, es decir, cómo debe ser el sistema. Generalmente corresponden a requisitos que son restricciones que se deben cumplir y que se encuentran fuera del conjunto de los requisitos funcionales. A continuación en la tabla 4 se especifican los requisitos no funcionales del sistema.

Tabla 4. Requisitos No Funcionales.

Ref. #	Función
FRQ-014	Interfaz del sistema debe ser amigable con el usuario.
FRQ-015	Bajo tráfico de datos.
FRQ-016	Portabilidad.
FRQ-017	La aplicación móvil debe ser desarrollada para Android.

4.4.2.1. Casos de Uso Extendido del sistema

Tabla 5. Caso de uso extendido “Agregar Servicios”.

Caso de uso:	Agregar Servicios (CU01).	
Actores:	Administrador.	
Propósito:	Agregar servicios al sistema.	
Resumen:	El Administrador agrega todos los servicios que brinda el centro de atención, a la base de datos ubicada en el servidor central.	
Tipo:	Primario y esencial.	
Precondiciones:	■ La base de datos, servidor web y servidor central deben estar operativos.	
Curso normal de los eventos		
Acción de los actores		Respuesta del sistema
<p>1. Este caso de uso comienza cuando se debe preparar el sistema para ser utilizado en el centro de atención.</p> <p>2. El Administrador se dirige a la opción que le permite agregar un servicio.</p> <p>4. El Administrador ingresa el nombre del servicio y agrega el servicio al sistema.</p>		<p>3. Muestra la interfaz para realizar esta acción.</p> <p>5. La base de datos recibe y graba la información.</p> <p>6. El servicio es agregado y se muestra en la ventana.</p>
Cursos alternativos		
<p>4. El servicio ya existe.</p> <p>5. El sistema no responde ante el envío de los datos.</p>		
Post condiciones:	El sistema queda esperando para repetir el proceso o realizar uno diferente.	

Tabla 6. Caso de uso extendido “Agregar Módulo Atención”.

Caso de uso:	Agregar Módulo Atención (CU02).
Actores:	Administrador.
Propósito:	Agregar módulos de atencion al sistema.
Resumen:	El Administrador agrega tal sistema todos los módulos donde se atenderán clientes.
Tipo:	Primario y esencial.
Precondiciones:	■ La base de datos, servidor web y servidor central deben estar operativos.
Curso normal de los eventos	
Acción de los actores	Respuesta del sistema
1. Este caso de uso comienza luego de que el Administrador haya ingresado los servicios (CU01). 2. El Administrador se dirige a la opción que le permite agregar un módulo de atención.  4. El Administrador ingresa el nombre del módulo de atención y lo agrega al sistema.	3. Muestra la interfaz para realizar esta acción.  5. La base de datos recibe y graba la información. 6. El módulo de atención es agregado y se muestra en la ventana.
Cursos alternativos	
4. El módulo de atención ya existe. 5. El sistema no responde ante el envío de los datos.	
Post condiciones:	El sistema queda esperando para repetir el proceso o realizar uno diferente.

Tabla 7. Caso de uso extendido “Seleccionar Servicio”.

Caso de uso:	Seleccionar Servicio (CU03).		
Actores:	Cliente.		
Propósito:	Obtener un turno de atención.		
Resumen:	El usuario se abre la aplicación <b>Mi Turno</b> y presiona un servicio, mostrado en la pantalla principal, para obtener un turno de atención.		
Tipo:	Primario y esencial.		
Precondiciones:	<ul style="list-style-type: none"><li>■ El dispositivo móvil debe tener conexión a Internet.</li><li>■ La base de datos, servidor web y servidor central deben estar operativos.</li></ul>		
Curso normal de los eventos			
Acción de los actores		Respuesta del sistema	
<p>1. Este caso de uso comienza cuando un Cliente se acerca a un centro de atención y, desea obtener un turno para realizar un trámite en algún servicio del centro de atención.</p> <p>2. El Cliente presiona un botón correspondiente al servicio que desea.</p>		<p>3. Muestra un cuadro de diálogo donde pregunta <b>¿Desea obtener ticket de atención?</b>.</p> <p>5. Muestra un mensaje confirmando la acción.</p> <p>6. Muestra una nueva pantalla donde se indica el <b>Turno</b> y <b>Servicio</b>.</p>	
Cursos alternativos			
<p>4. El Cliente presionó equivocadamente el servicio y presiona el botón <b>NO</b>.</p> <p>5. El dispositivo móvil pierde comunicación con el servidor central.</p>			
Post condiciones:	La aplicación móvil queda trabajando en segundo plano a la espera de recibir Notificaciones.		

4.5.   Diseño

4.5.1.   Casos de Uso Real

Tabla 8. Caso de uso Real “Agregar Servicios”.

Caso de uso:	Agregar Servicios (CU01).
Actores:	Administrador.
Propósito:	Agregar servicios al sistema.
Resumen:	El Administrador agrega todos los servicios que brinda el centro de atención, a la base de datos ubicada en el servidor central.
Tipo:	Primario y esencial.
Precondiciones:	■ La base de datos, servidor web y servidor central deben estar operativos.
Curso normal de los eventos	
Acción de los actores	Respuesta del sistema
1. Este caso de uso comienza cuando se debe preparar el sistema para ser utilizado en el centro de atención. 2. El Administrador se dirige a la opción Configuración (A), luego a Agregar Servicios (B).  4. El Administrador ingresa el nombre del servicio en la entrada de texto (C) y presiona el boton Agregar (D).	3. Muestra la interfaz para realizar esta acción.  5. La base de datos recibe y graba la información. 6. El servicio es agregado y se muestra en la lista de servicios (E).
Cursos alternativos	
4. El servicio ya existe. 5. El sistema no responde ante el envío de los datos.	
Post condiciones:	El sistema queda esperando para repetir el proceso o realizar uno diferente.

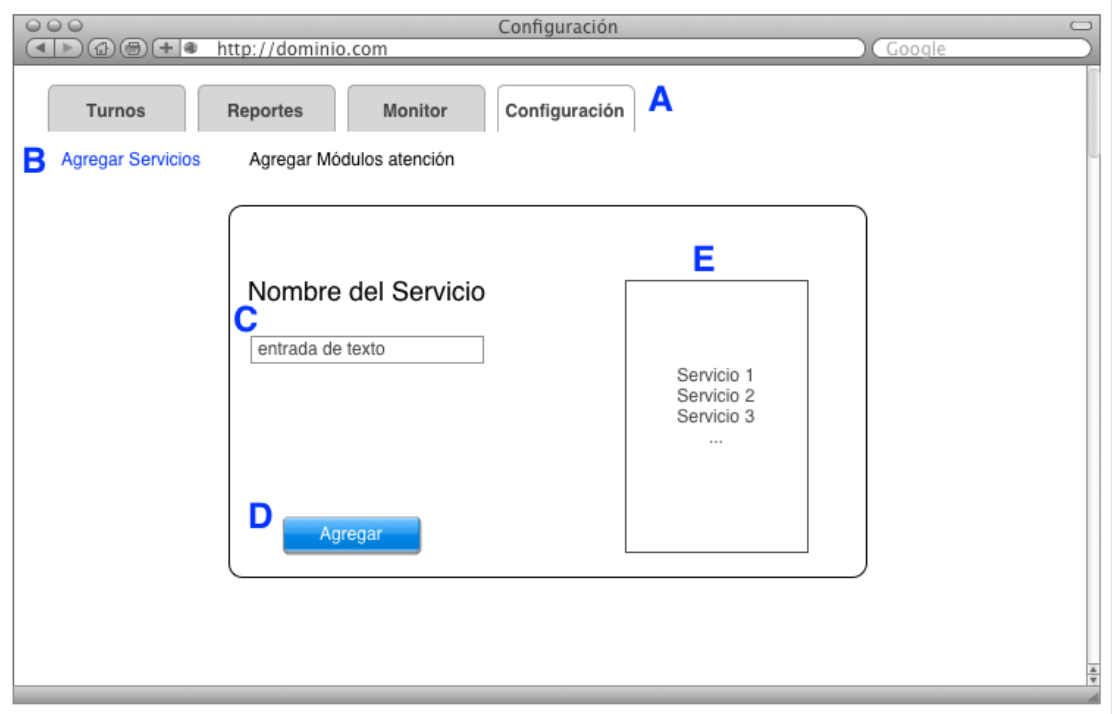


Figura 13. *Mockup* Caso de Uso 1 (CU01).

Tabla 9. Caso de uso Real “Agregar Módulo Atención”.

Caso de uso:	Agregar Módulo Atención (CU02).
Actores:	Administrador.
Propósito:	Agregar módulos de atencion al sistema.
Resumen:	El Administrador agrega tal sistema todos los módulos donde se atenderán clientes.
Tipo:	Primario y esencial.
Precondiciones:	■ La base de datos, servidor web y servidor central deben estar operativos.
Curso normal de los eventos	
Acción de los actores	Respuesta del sistema
1. Este caso de uso comienza luego de que el Administrador haya ingresado los servicios (CU01). 2. El Administrador se dirige a la opción <b>Configuración (A)</b> , luego a <b>Agregar Servicios (B)</b> .  4. El Administrador ingresa el nombre del módulo de atención en la entrada de texto (C) y presiona el boton <b>Agregar (D)</b> .	3. Muestra la interfaz para realizar esta acción.  5. La base de datos recibe y graba la información. 6. El módulo de atención es agregado y se muestra en la lista de servicios (E)
Cursos alternativos	
4. El módulo de atención ya existe. 5. El sistema no responde ante el envío de los datos.	
Post condiciones:	El sistema queda esperando para repetir el proceso o realizar uno diferente.

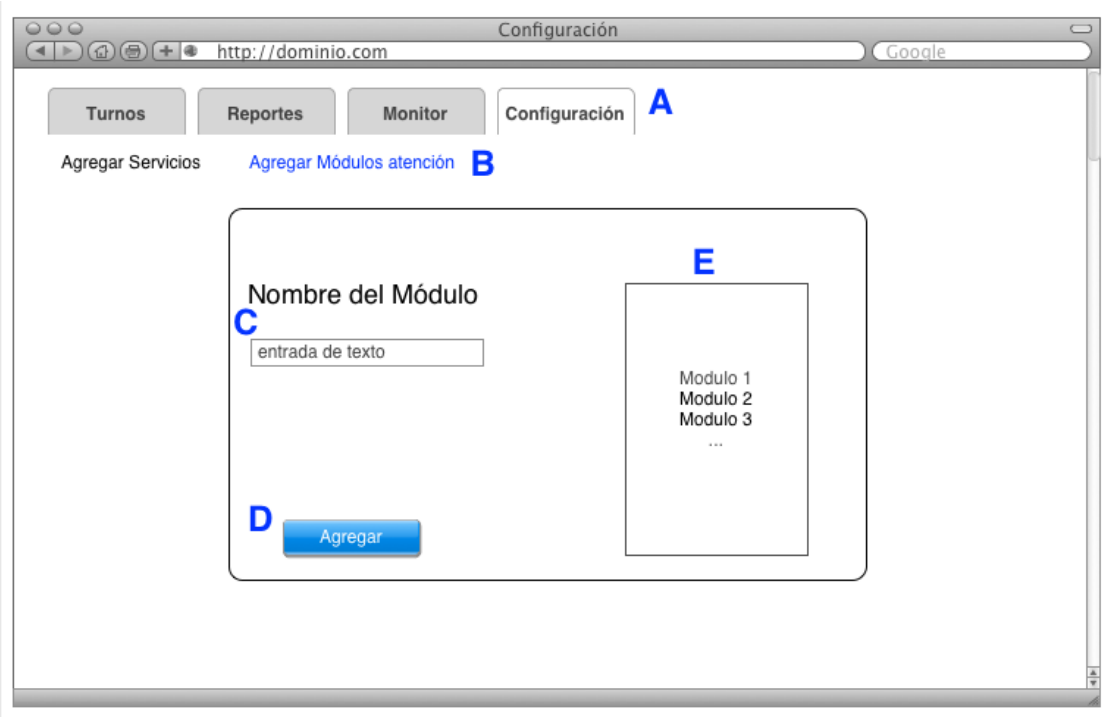


Figura 14. *Mockup* Caso de Uso 2 (CU02).



Tabla 10. Caso de uso Real “Seleccionar Servicio”.

Caso de uso:	Seleccionar Servicio (CU03).		
Actores:	Cliente.		
Propósito:	Obtener un turno de atención.		
Resumen:	El usuario se abre la aplicación <b>Mi Turno</b> y presiona un servicio, mostrado en la pantalla principal, para obtener un turno de atención.		
Tipo:	Primario y esencial.		
Precondiciones:	<ul style="list-style-type: none"><li>■ El dispositivo móvil debe tener conexión a Internet.</li><li>■ La base de datos, servidor web y servidor central deben estar operativos.</li></ul>		
Curso normal de los eventos			
Acción de los actores		Respuesta del sistema	
<p>1. Este caso de uso comienza cuando un Cliente se acerca a un centro de atención y, desea obtener un turno para realizar un trámite en algún servicio del centro de atención.</p> <p>2. El Cliente presiona un botón correspondiente al servicio que desea (A).</p> <p>4. El Cliente en caso que desee ese servicio presiona el botón <b>SI</b> (C).</p>		<p>3. Muestra un cuadro de diálogo donde pregunta <b>¿Desea obtener ticket de atención?</b> (B).</p> <p>5. Muestra un mensaje confirmando la acción.</p> <p>6. Muestra una nueva pantalla donde se indica el <b>Turno</b> y <b>Servicio</b> (D).</p>	
Cursos alternativos			
<p>4. El Cliente presionó equivocadamente el servicio y presiona el botón <b>NO</b>.</p> <p>5. El dispositivo móvil pierde comunicación con el servidor central.</p>			
Post condiciones:	La aplicación móvil queda trabajando en segundo plano a la espera de recibir Notificaciones.		



Figura 15. *Mockup* Caso de Uso 3 (CU03).

4.5.2. Diagramas de Secuencia

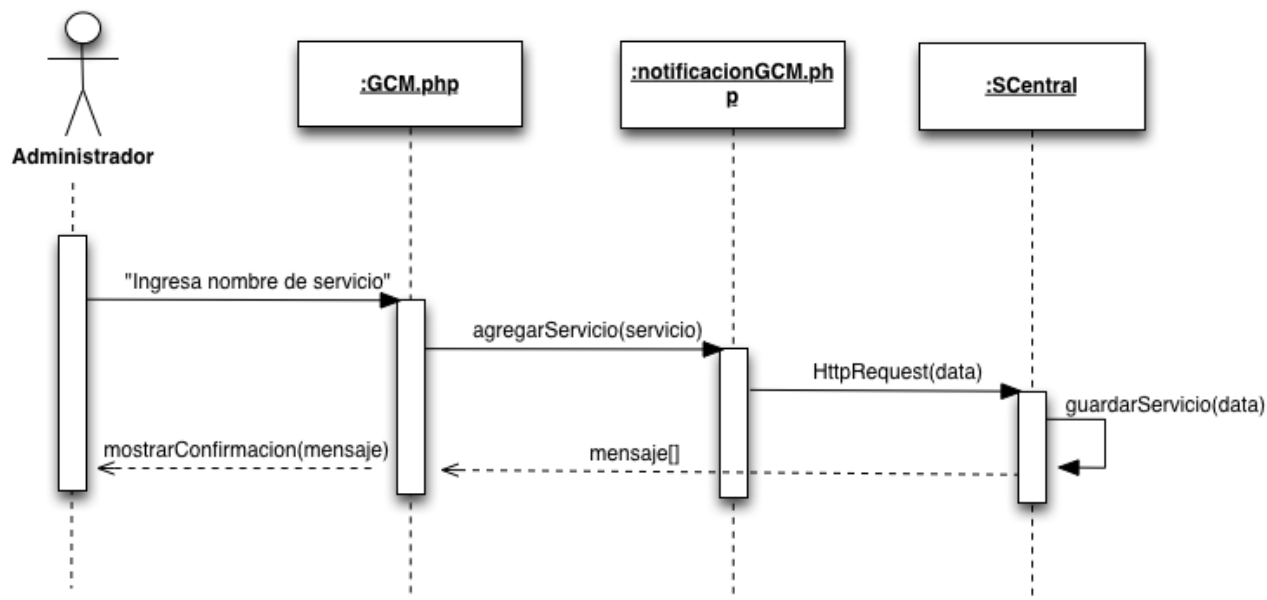


Figura 16. Diagrama de Secuencia Caso de Uso Agregar Servicios.

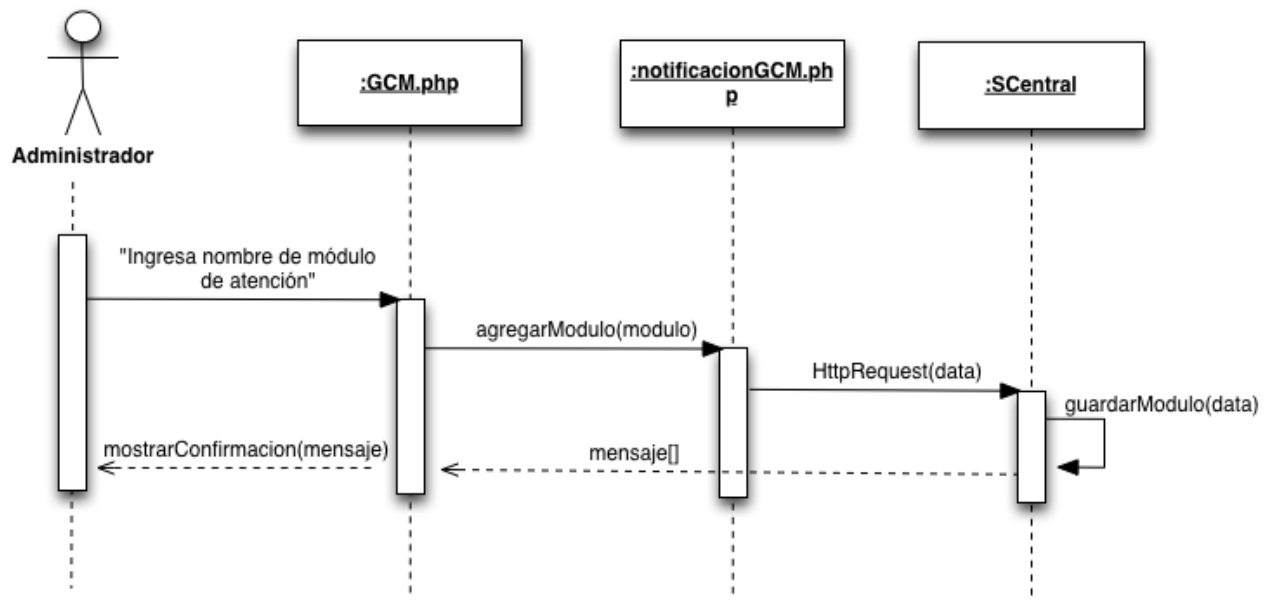


Figura 17. Diagrama de Secuencia Caso de Uso Agregar Módulo Atención.

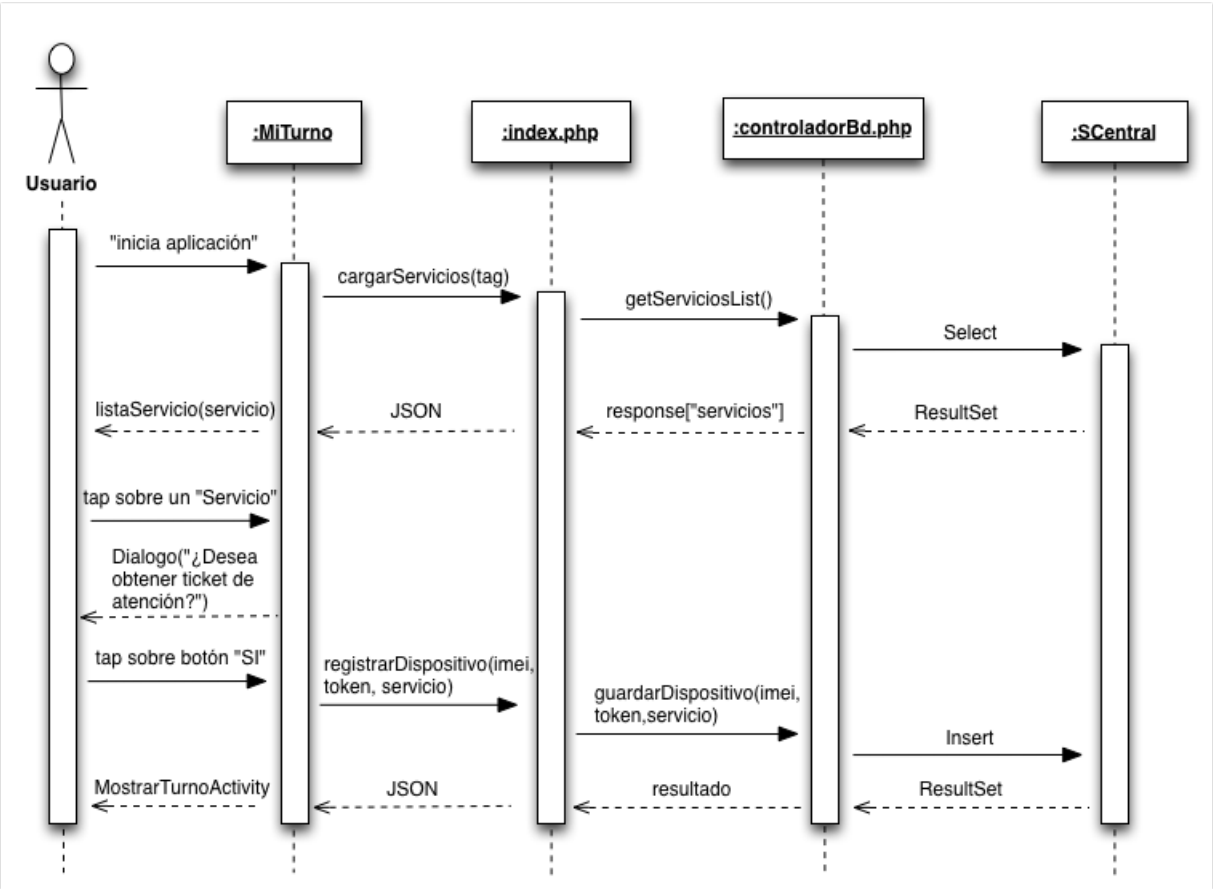


Figura 18. Diagrama de Secuencia Caso de Uso Seleccionar Servicio.

4.5.3. Diagrama de Clases

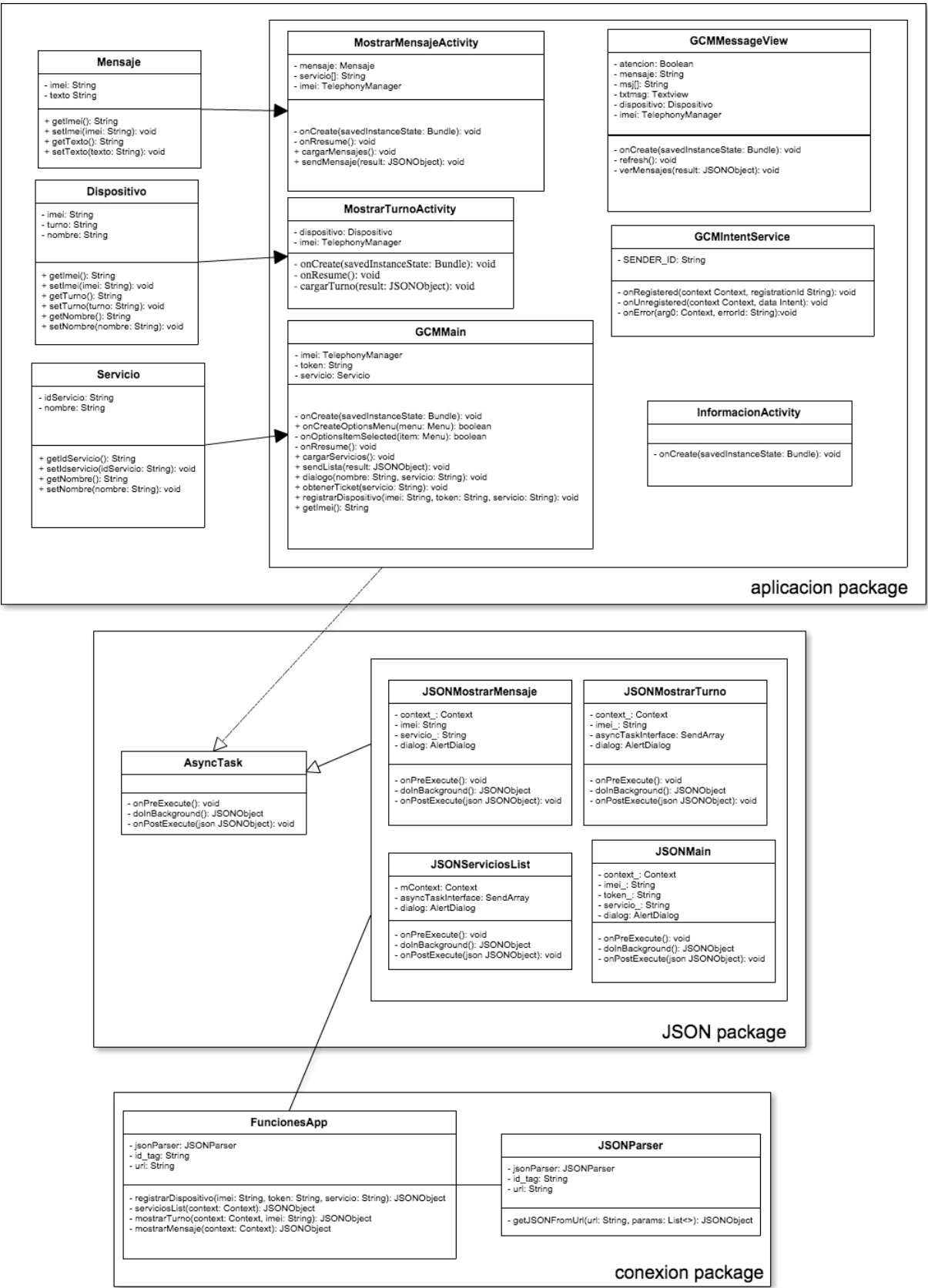


Figura 19. Diagrama de Clases Aplicación Móvil.

4.5.4. Diagrama de Procesos

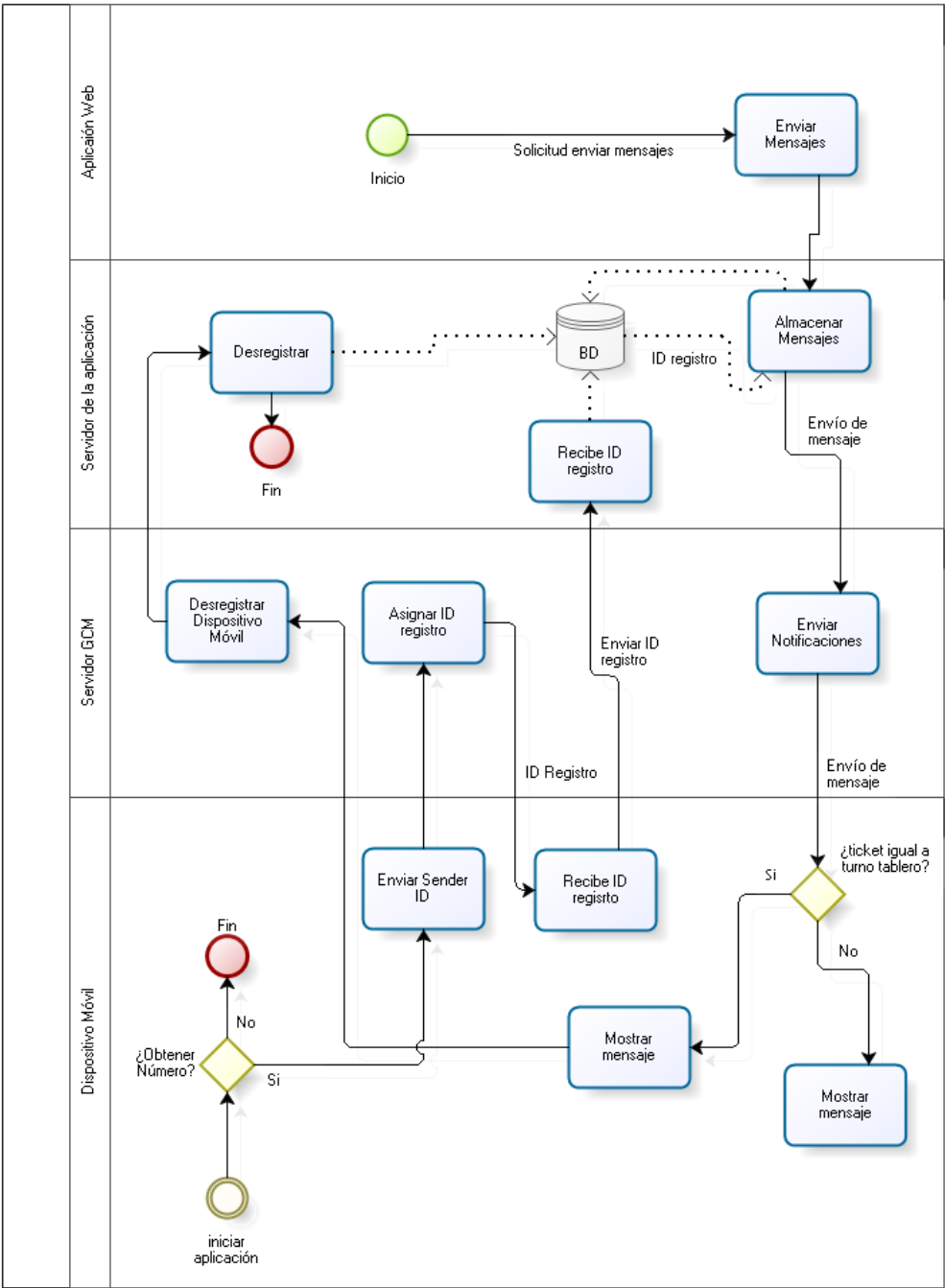


Figura 20. Diagrama de Proceso Aplicación Móvil.

4.5.5. Modelo de Datos

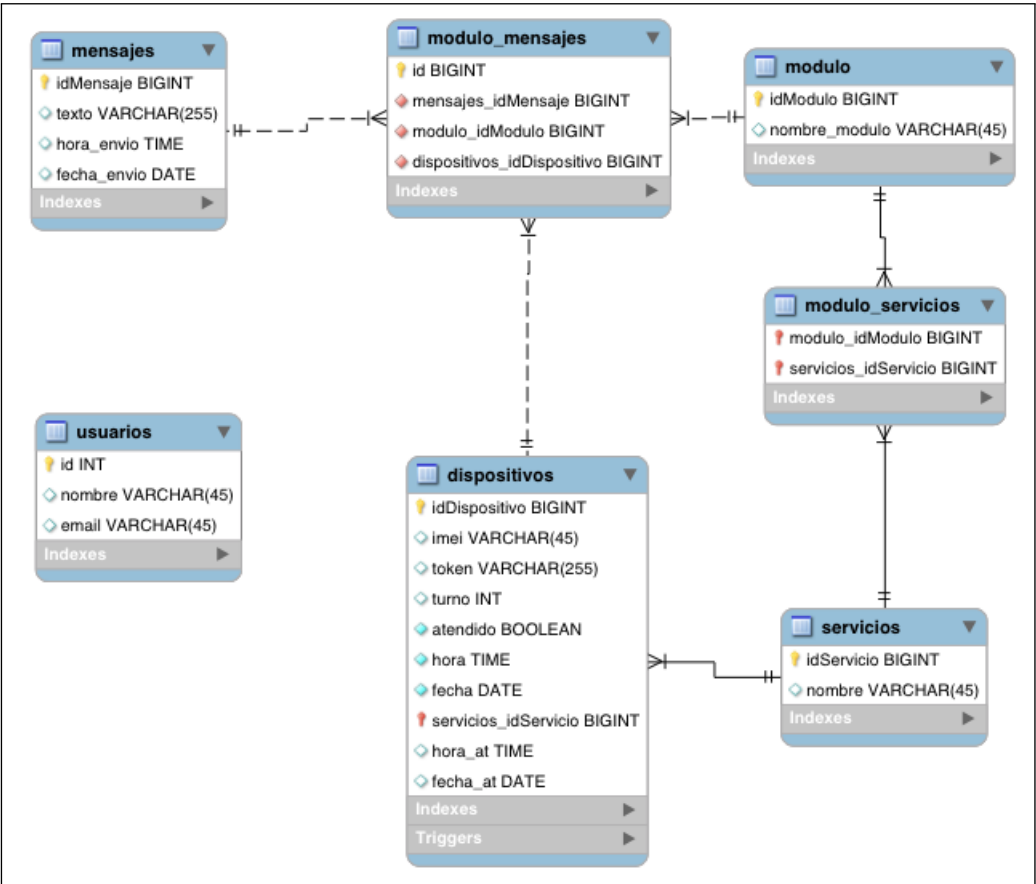


Figura 21. Modelo de datos.

4.5.5.1. Diccionario de datos

Tabla 11. Diccionario de datos tabla “dispositivos”.

Columna	Tipo de dato	Descripción
idDispositivos	BIGINT	Identificador único de cada dispositivo.
imei	VARCHAR(45)	Identificador propio de cada <i>smartphone</i> .
token	VARCHAR(255)	Identificador entregado por GCM a cada dispositivo móvil registrado en el sistema.
turno	INT	Número de turno de atención asignado por el sistema a un dispositivo móvil.
atendido	BOOLEAN	Estado de atención en que encuentra un dispositivo, TRUE o FALSE.
hora	TIME	Hora en que el dispositivo solicita un turno de atención.
fecha	DATE	Fecha en que el dispositivo solicita un turno de atención.
servicios_idServicio	BIGINT	Clave foránea de la tabla servicios.
hora_at	TIME	Hora en que el cliente es llamado para ser atendido en un módulo de atención.
fecha_at	DATE	Fecha en que el cliente es llamado para ser atendido en un módulo de atención.

Tabla 12. Diccionario de datos tabla “mensajes”.

Columna	Tipo de dato	Descripción
idMensaje	BIGINT	Identificador único de cada mensaje.
texto	VARCHAR(255)	Mensaje enviado desde el sistema a cada <i>smartphone</i> .
hora_envio	TIME	Hora en que se envió el mensaje.
fecha_envio	DATE	Fecha en que se envió el mensaje.



Tabla 13. Diccionario de datos tabla “modulo”.

Columna	Tipo de dato	Descripción
idModulo	BIGINT	Identificador único de cada módulo de atención.
nombre_modulo	VARCHAR(45)	Nombre del módulo de atención.

Tabla 14. Diccionario de datos tabla “servicios”.

Columna	Tipo de dato	Descripción
idServicio	BIGINT	Identificador único de cada servicio.
nombre	VARCHAR(45)	Nombre del servicio.

Tabla 15. Diccionario de datos tabla “usuarios”.

Columna	Tipo de dato	Descripción
id	INT	Identificador único de cada usuario.
nombre	VARCHAR(45)	Nombre y apellido del usuario.
email	VARCHAR(45)	Correo electrónico del usuario.

Tabla 16. Diccionario de datos tabla “modulo\_mensajes”.

Columna	Tipo de dato	Descripción
id	BIGINT	Identificador único de la tabla.
mensajes_idMensaje	BIGINT	Identificador único de cada mensaje.
modulo_idModulo	BIGINT	Identificador único de cada módulo de atención.
dispositivos_idDispositivos	BIGINT	Identificador único de cada dispositivo.

Tabla 17. Diccionario de datos tabla “modulo\_servicios”.

Columna	Tipo de dato	Descripción
modulo_idModulo	Identificador único de cada	módulo de atención.
servicios_idServicios	BIGINT	Identificador único de cada servicio.

## 5. IMPLEMENTACIÓN DEL PROTOTIPO

### 5.1. Configuración ambiente de desarrollo y problemas enfrentados

Este capítulo describe las diferentes etapas por las que ha atravesado el desarrollo de este sistema hasta llegar a su última iteración. Se inicia con una breve reseña de los pasos previos al desarrollo, es decir, la instalación de herramientas y configuración de la máquina de trabajo donde se desarrollará la aplicación móvil y web. Luego, a modo de introducción, se da paso a la presentación de una aplicación básica utilizando el IDE *Android Studio*, posteriormente se muestran los cuatro prototipos del sistema señidos a la metodología de desarrollo de *software* elegida.

Las características del equipo de trabajo y dispositivo móvil para la realización y pruebas del sistema fueron:

- *Notebook MacBook Pro*, procesador 2,5 Ghz *Intel Core i5*.
- *Smartphone* Motorola Moto X, con Android 4.4.4.

Fue necesario instalar y activar algunos servicios que fueron escogidos para el proyecto. En la tabla 18 se indican las herramientas instaladas, activadas y/o servicios iniciados, junto con su versión, en la máquina de trabajo.

Tabla 18. Herramientas, estado y versión en máquina de trabajo.

Herramienta	Acción requerida	Versión
Apache	Iniciar servicio	2.4.9
PHP	Activar servicio	5.5.14
Android Studio	Descargar e instalar	1.0
MySQL	Descargar, instalar e iniciar	5.6.23
MySQLWorkbench	Descargar e instalar	6.1

Las herramientas que requerieren ser iniciadas y activadas vienen instaladas previamente en el Sistema Operativo. En cambio, las herramientas que no están incluidas fueron descargadas de su propio sitio web e instaladas posteriormente.

Durante instalación de las herramientas mencionadas no hubo mayores problemas; sin embargo, en los servicios, expuestos en la tabla anterior, se activaron e iniciaron mediante el uso de comandos a través del terminal de OS X.

Para iniciar Apache, abrir el terminal e ingresar en modo super usuario, *root*, para tener todos los permisos. En la Figura 22 se exhiben los comandos:

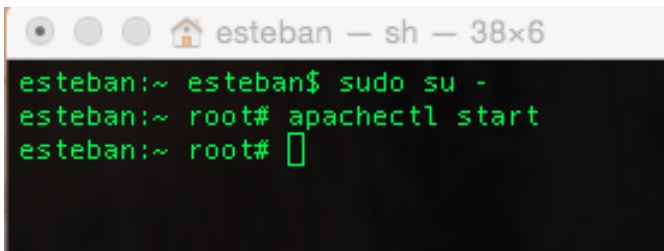


Figura 22. Iniciar Apache desde el terminal de OS X.

Seguido de lo anterior, es recomendable corroborar que se ha iniciado Apache. Para ello, abrir el navegador web y en la barra de direcciones escribir: 'http://localhost', debe aparecer el mensaje 'It works!' como muestra la Figura 23.

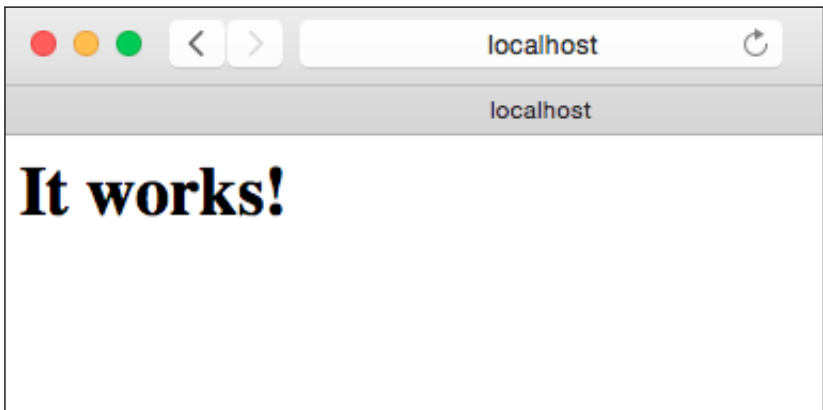
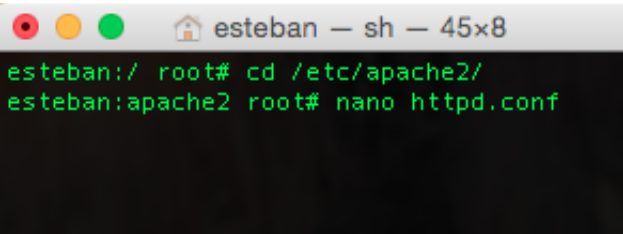


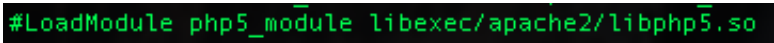
Figura 23. Servicio Apache funcionando correctamente.

Posteriormente, se continúa con la activación de PHP. Nuevamente usando el terminal, dirigirse a la carpeta apache2: 'cd /etc/apache2/'. En esa carpeta se encuentra el archivo 'http.conf', dentro del archivo descomentar, borrar el signo '#', la siguiente línea: 'LoadModule php5\_module libexec/apache2/libphp5.so'. Por último reiniciar Apache: 'apachectl restart'. Las figuras 24, 25 y 26 muestran lo descrito anteriormente.



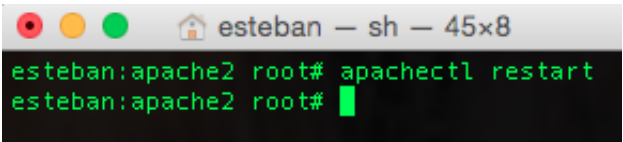
```
esteban — sh — 45x8
esteban:/ root# cd /etc/apache2/
esteban:apache2 root# nano httpd.conf
```

Figura 24. Acceso a carpeta apache2 y edición archivo httpd.conf.



```
#LoadModule php5_module libexec/apache2/libphp5.so
```

Figura 25. Habilitar PHP para Apache.



```
esteban:apache2 root# apachectl restart
esteban:apache2 root#
```

Figura 26. Reiniciando Apache.

Para finalizar, es importante aclarar que MySQLWorkbench<sup>4</sup>, Android Studio<sup>5</sup> y MySQL<sup>6</sup> fueron descargados e instalados sin inconvenientes.

**5.2. Presentación de prototipos**

En esta sección, se muestra el crecimiento y nuevas funciones que ha logrado el sistema en las diferentes iteraciones. En cada caso se entrega una explicación de los aspectos que se consideran más importantes, y que forman parte de situaciones que exigieron interiorizarse para superar incidencias que surgieron en algunas de las etapas que se enseñan en los siguientes puntos.

**5.2.1. Prototipo 1: Aplicación básica en android Studio**

La primera aplicación se realizará a modo de ejercicio para corroborar que está funcionando correctamente *Android Studio* y el *smartphone*, para probar las aplicaciones hechas. Antes de realizar alguna acción es necesario tener el celular en modo depuración. Para esto hay que dirigirse a: Configurar ->Programador ->Depuración por USB. La

<sup>4</sup>Es una herramienta visual de diseño, administración, creación y mantención de bases de datos MySQL. Más información en: <http://www.mysql.com/products/workbench/>  
<sup>5</sup><http://developer.android.com/sdk/index.html>  
<sup>6</sup><http://dev.mysql.com/downloads/mysql/>

Figura 27 muestra la opción que debe estar seleccionada.

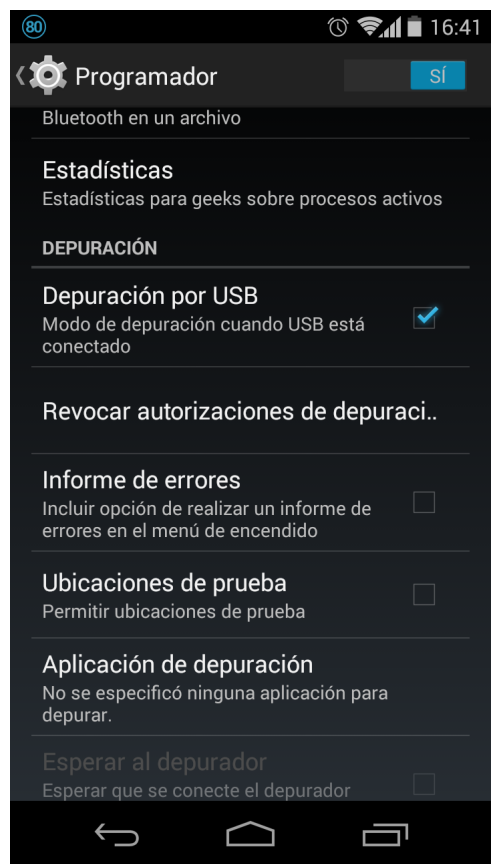


Figura 27. Modo Depuración por USB activado en *Android* 4.4.4.

En la Figura 28, se puede apreciar la ventana de bienvenida de *Android Studio*. Aquí es posible crear nuevos proyectos, abrir proyectos existentes, entre otras funciones.

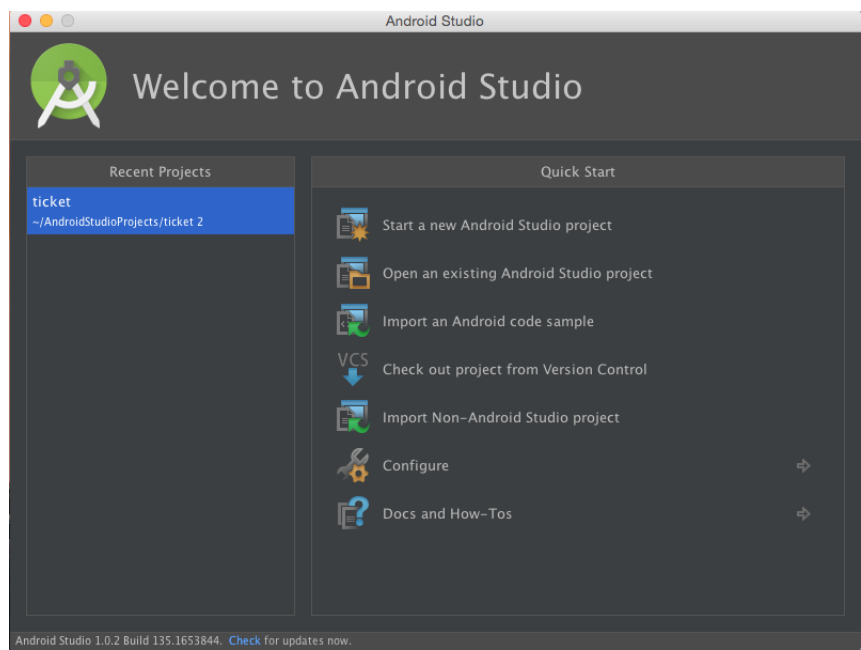


Figura 28. Ventana bienvenida *Android Studio*.

En el momento que se crea un nuevo proyecto, luego de ingresar el nombre que éste tendrá, el asistente solicita elegir la versión mínima sobre la cual correrá nuestra aplicación móvil. Por lo tanto, es importante elegir correctamente, para evitar futuras complicaciones de compatibilidad cuando se cargue la aplicación en equipos que, posiblemente, tengan una versión menor a la elegida. En el menú desplegable de la Figura 29 se debe elegir una versión de API que será la versión mínima que necesitará la aplicación para ejecutarse en un *smartphone*. La versión más antigua disponible en el menú es *Android 4.0.3 - IceCreamSandwich* y la más nueva es la versión reciente *Android 5.0 - Lollipop*.

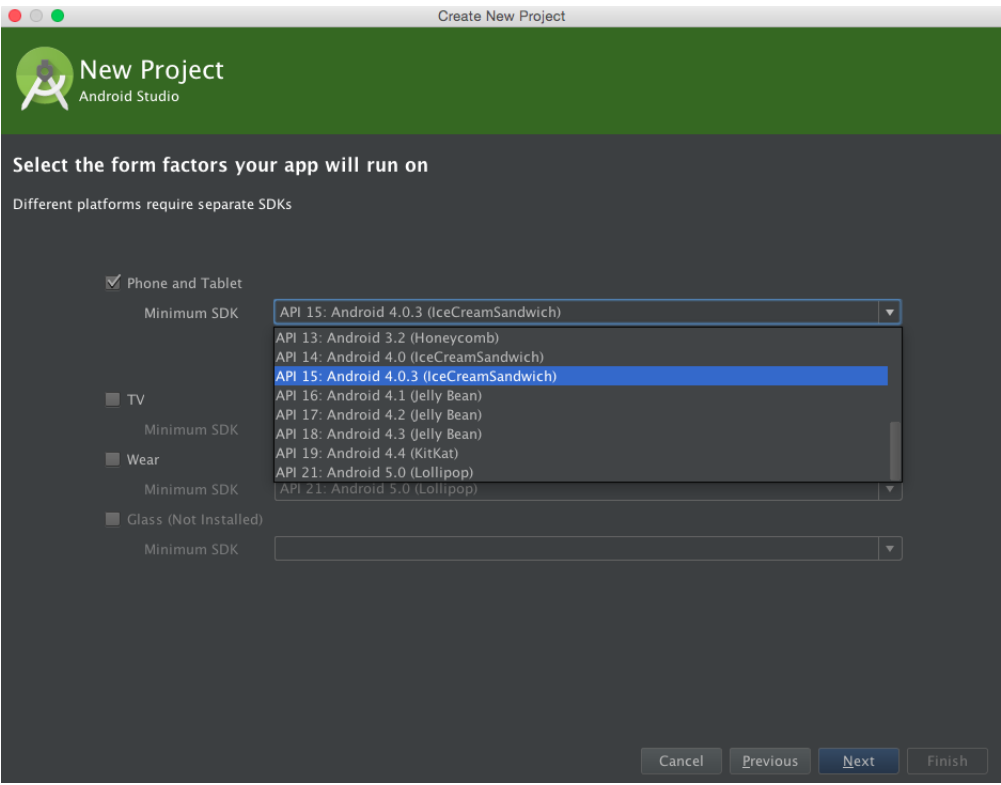


Figura 29. Elección de API mínima requerida.

El proyecto en el entorno de desarrollo, Figura 30, a la izquierda cuenta con un árbol para navegar y acceder a las diversas carpetas y archivos que contiene la aplicación. El código fuente de nuestro programa *Android*, **controlador**, se encuentra dentro de la carpeta: app ->src ->main ->java. Es ahí donde se encuentran las clases del proyecto que dan vida a la interfaz gráfica de la aplicación, que en el patrón MVC corresponde a la **vista**. Ésta última, dentro del proyecto la ubicamos en la ruta: app ->src ->main ->res ->layout. Acá están almacenadas todas las pantallas que tendrá nuestra aplicación. A la derecha de la venta en la Figura 30 se aprecia el código fuente de la aplicación básica realizada con fines de prueba.



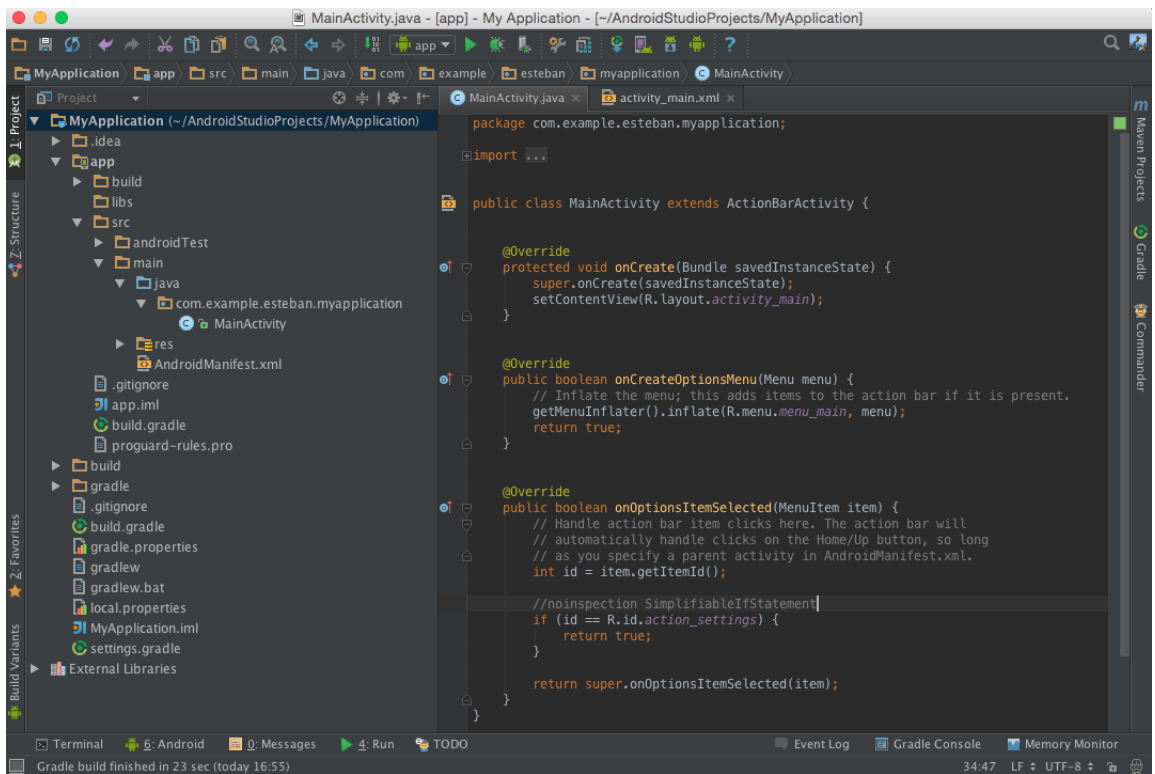


Figura 30. Proyecto nuevo en *Android Studio*.

Luego de proceder a compilar el proyecto, y previamente haber conectado el celular al computador de trabajo, la aplicación móvil es enviada al smartphone a través del cable USB. Ya en el dispositivo ésta es instalada y ejecutada automáticamente. En la Figura 31 se ve el resultado del proyecto creado anteriormente. Debido a que no hubo inconvenientes en esta pequeña prueba, se concluye que el entorno de trabajo está listo para iniciar el desarrollo de la aplicación móvil, que forma parte del sistema propuesto como solución a la problemática antes definida en la sección 3.1 del documento.

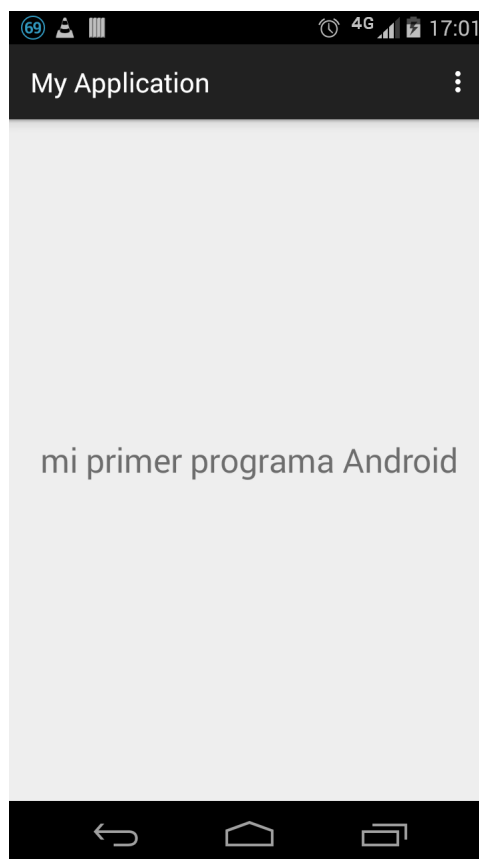


Figura 31. Primera aplicación móvil de prueba.

### 5.2.2. Prototipo 2: Cliente y Servidor GCM

En esta sección, se mostrará la implementación del servidor y cliente que envía y recibe mensajes, respectivamente, a través de *Google Cloud Messaging*.

#### 5.2.2.1. Cliente

Antes de crear un proyecto en *Android Studio* es necesario crear un proyecto, ver Figura 32, en *Google Developers Console*.<sup>7</sup>

---

<sup>7</sup>Es un espacio que tienen los desarrolladores de *Google* para la gestión y visualización de las API de la compañía que sus proyectos utilizan. Más información en: <https://developers.google.com/console/help/new/>

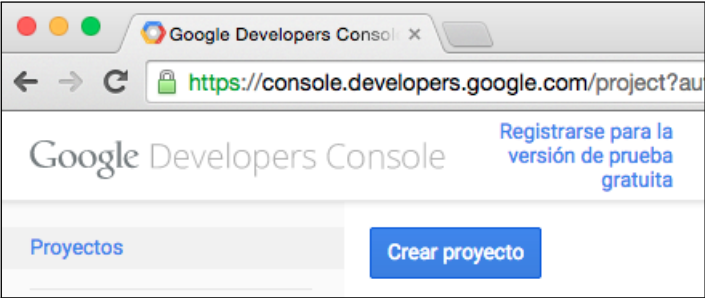


Figura 32. Crear nuevo proyecto en Google Developers Console.

Luego de ser creado el proyecto, se despliega un resumen con información del proyecto (Figura 33), en la que aparece el *PROJECT NUMBER* que corresponde a un identificador único de los proyectos creados en esta plataforma. Mas adelante será incluido en la aplicación móvil.

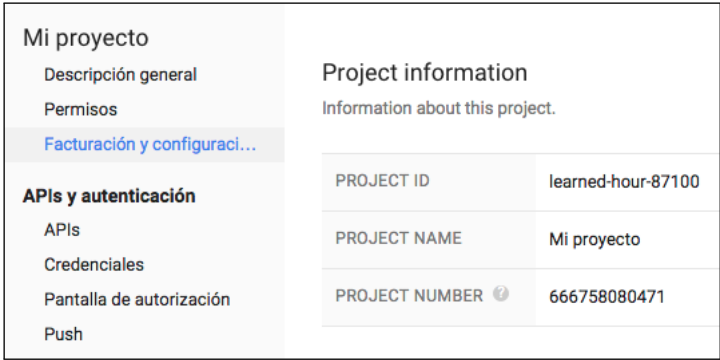


Figura 33. Resumen proyecto.

Posteriormente es necesario habilitar APIs que provee *Google* para los diferentes servicios que ofrece (*Calendar, Contacts, Gmail, Google Maps Android, Google Play Game Services*, entre muchos servicios más.). Para efectos del proyecto sólo se habilitará la API *Google Cloud Messaging for Android* que se puede apreciar en la Figura 34.

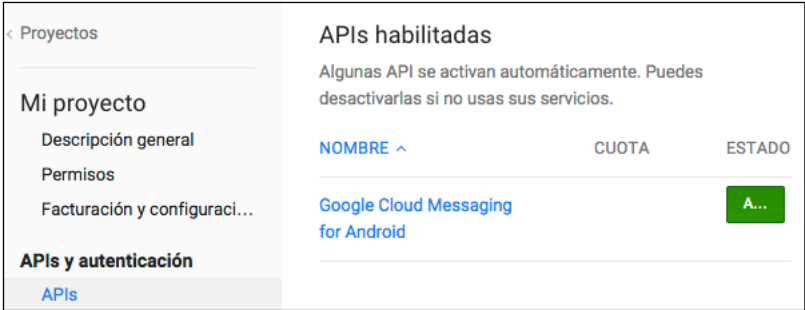


Figura 34. Activación de API para proyecto.

Dentro de la API seleccionada, es importante crear una clave nueva para un servidor a

través del cual circulan las notificaciones que sean enviadas desde el servidor del sistema. Ver Figuras 35, 36 y 37. En la Figura 38 se muestra un resumen con información a cerca del servidor creado. En la primera línea del resumen se encuentra la Clave DE LA API, que corresponde a un identificador único de dicho servidor. Este identificador al igual que el *PROJECT NUMBER* serán incluidos en el sistema.

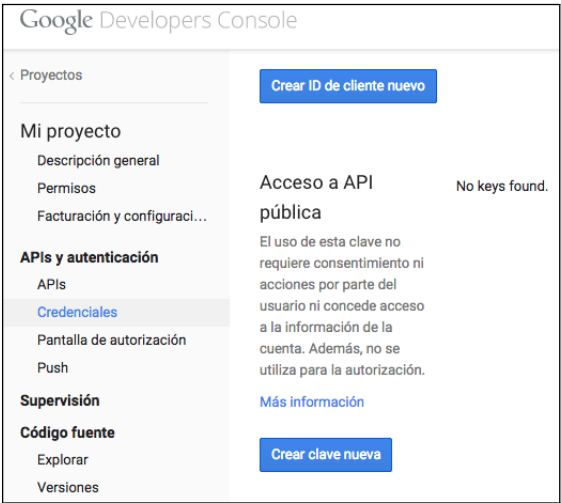


Figura 35. Crear clave nueva.



Figura 36. Crear clave nueva clave de servidor.

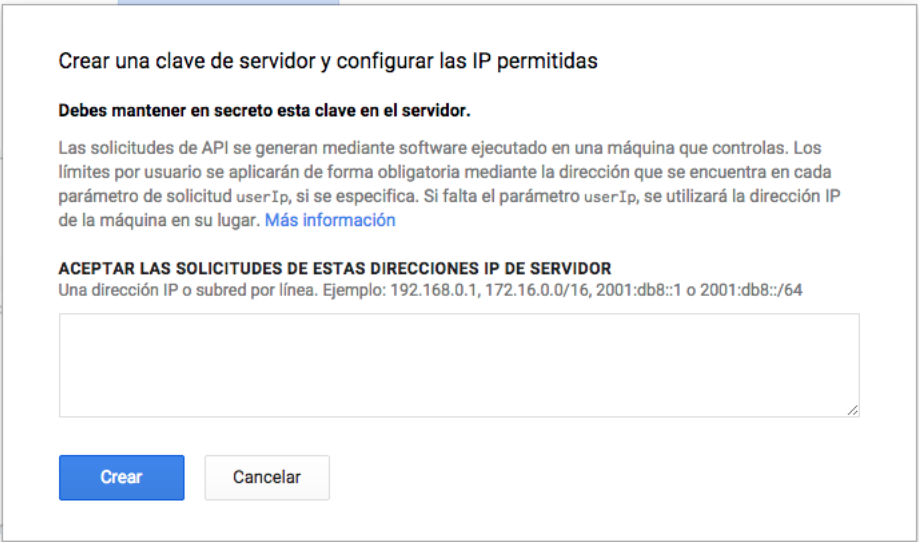


Figura 37. Crear clave de servidor.

Acceso a API pública

El uso de esta clave no requiere consentimiento ni acciones por parte del usuario ni concede acceso a la información de la cuenta. Además, no se utiliza para la autorización.

Más información

Crear clave nueva

Clave para las aplicaciones de servidor

CLAVE DE LA API	AlzaSyBQtCjwrd4OAL9GkEHDmeDflv4rl5Ms3xl
IPS	Se permite cualquier IP
FECHA DE ACTIVACIÓN	28 de feb. de 2015 13:45:00
ACTIVADO POR	es.orellana@gmail.com (tú)

Editar IP permitidas

Volver a generar la clave

Eliminar

Figura 38. Resumen servidor.

Dentro de *Android Studio* en el *SDK Manager* hay que instalar el *package* de *Google Play Services*, Figura 39, que contiene la librería *GCM.jar* que debe ser agregada al proyecto para que funcione la comunicación entre la aplicación *Android* y los servidores de *Google* desde donde vienen las notificaciones enviadas desde el servidor local.

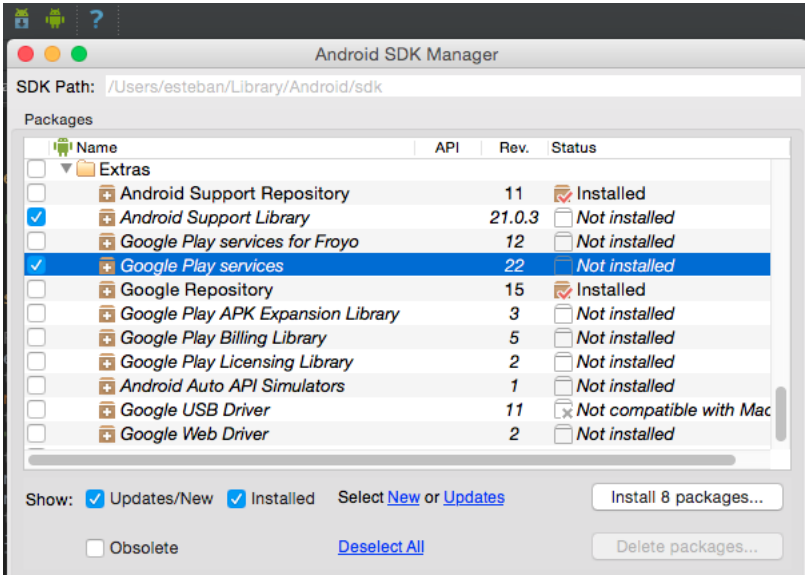


Figura 39. Instalación *Google Play Services*.

En el proyecto de la aplicación cliente para *Android* se tienen tres clases escritas en lenguaje java: *GCMMainActivity*, *GCMIntentService* y *GCMMessageView*. Entre todas se encargan de hacer posible: el registro del dispositivo. Es decir, solicitar un identificador único para el dispositivo móvil a los servidores de *GCM* y de esta forma poder recibir y visualizar las notificaciones. Lo anterior se muestra en la Figura 40.

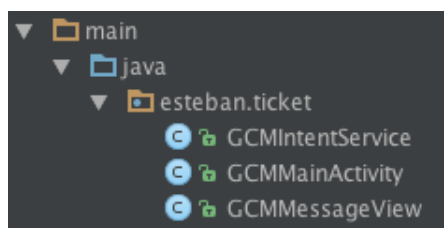


Figura 40. Clases del proyecto.

Como se mencionó anteriormente, el *PROJECT ID* sería incluido en el proyecto de *Android Studio*, Figura 41. De no ser agregado este ID, la aplicación móvil queda incomunicada y sería imposible que reciba los mensajes, ya que 'no sabe' de qué proyecto debe 'escuchar' los mensajes.

```
public class GCMIntentService extends GCMBaseIntentService {  
    // Se debe incluir el PROJECT ID de Google API en SENDER_ID  
    public static final String SENDER_ID = "666758080471";  
}
```

Figura 41. Incluir el *PROJECT ID* en la aplicación.

La pantalla que se muestra en la Figura 42 corresponde a la aplicación creada, el botón 'Registrar Dispositivo' al ser presionado realiza la acción de solicitar el identificador asociado al dispositivo. Éste nuevo ID, sirve para asociar el celular al servidor de la API creados anteriormente.

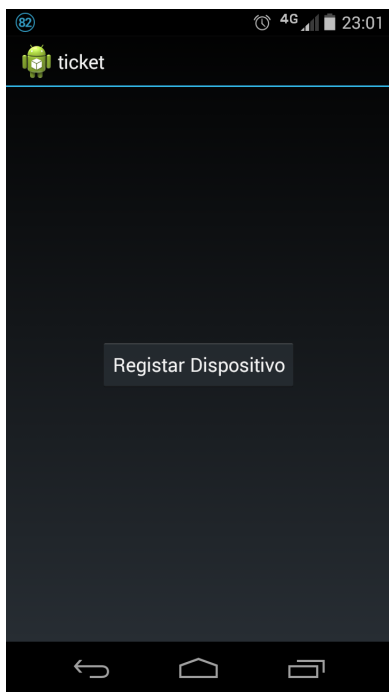


Figura 42. Registrar dispositivo desde la aplicación.

5.2.2.2. Servidor

El servidor está implementado en html y es la aplicación web que se mostró en las secciones anteriores. Su función es capturar datos en un formulario que posteriormente será enviado a una clase externa escrita en lenguaje PHP, la que se encarga de recibir los datos, encapsularlos y enviarlos al servidor de *Google Cloud Messaging* que se creó anteriormente. En la Figuras siguientes: 43, 44 y 45 se muestra el código del mismo y después se expone el servidor en el navegador web.

```
<html>
<head>
<title> Mensaje  Notificacion</title>
</head>
<body>
<form action="gcm_engine.php" method="post">
<br></br>
Ingrese mensaje de notificacion :
<br></br>
<input size=70% TYPE = "Text" VALUE="" NAME = "message">
<br></br>
<input type="submit" value="Enviar Mensaje"/>
</form>
</body>
</html>
```

Figura 43. Codigo html del Servidor.

```
<?php
// Mensaje a enviar
$message = $_POST['message'];

$apiKey = "AIzaSyCqAajpG-ff69TC_P3_zQ7zeoJ3NpXvwe8";

// direccion donde se enviarán los datos
$url = 'https://android.googleapis.com/gcm/send';

$fields = array(
    'registration_ids' => array($_POST['registrationIDs']),
    'data' => array( "message" => $message ),
);

$headers = array(
    'Authorization: key=' . $apiKey,
    'Content-Type: application/json'
);

// abrir conexión
$ch = curl_init();

curl_setopt( $ch, CURLOPT_URL, $url );
curl_setopt( $ch, CURLOPT_POST, true );
curl_setopt( $ch, CURLOPT_HTTPHEADER, $headers);
curl_setopt( $ch, CURLOPT_RETURNTRANSFER, true );
curl_setopt( $ch, CURLOPT_POSTFIELDS, json_encode( $fields ) );

// se envia la información
$result = curl_exec($ch);

// se cierra la conexión
curl_close($ch);

echo $result;
?>
```

Figura 44. Codigo clase gcm\_engine.php.



Figura 45. Servidor.

En la Figura 45, el servidor tiene tres campos de texto. El primero, corresponde a la clave API del servidor de *Google* antes creado y expuesto en la Figura 38. El segundo campo de texto, solicita que se ingrese el ID de registro del dispositivo. Éste es entregado, luego de presionar el botón 'Registrar Dispositivo' en la aplicación, ver Figura 42, por el servidor de la API creado en los pasos anteriores. En el tercer campo de texto, se debe ingresar el mensaje o notificación a enviar al o los dispositivos que tienen instalada la aplicación y se han registrado al presionar el botón antes mencionado. Por último, se debe presionar el boton 'Enviar Notificación' y el mensaje será enviado inmediatamente y podrá ser visualizado en el dispositivo móvil. Ver Figuras 46 y 47.

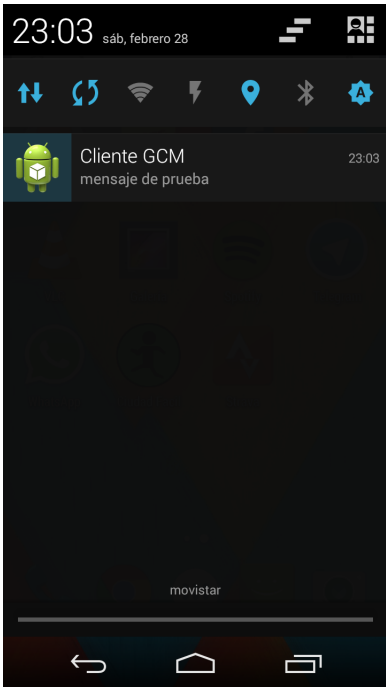


Figura 46. Notificación *Push* recibida.



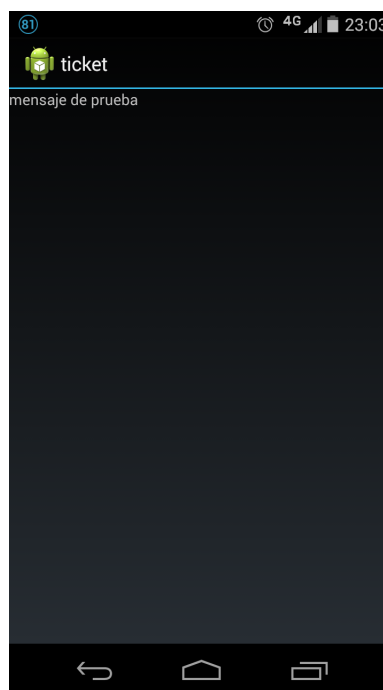


Figura 47. Visualizacion contenido de notificación.

En este ítem se ha explicado paso por paso como se implementó este primer prototipo. Además, se resuelve una parte esencial de todo el sistema propuesto como solución a la problemática definida.

### 5.2.3. Prototipo 3: Comunicar Android y MySQL

Una forma de comunicar una aplicación *Android* con una base de datos externa, que se encuentre en una máquina remota, es utilizando servicios web y un formato de intercambio de datos como es JSON<sup>8</sup>. En la sección 2.1.4 - *Web Service*, se explicó como el es flujo de datos entre: una aplicación *Android* - PHP - MySQL.

Por el lado de la aplicación *Android*, se agregaron funciones y clases al proyecto que realizan la tarea de capturar datos y enviarlos al servicio web 'index.php' ubicado en la estación de trabajo que a su vez juega el papel de servidor externo. En la Figura 48 se aprecia la misma pantalla de la aplicación pero, ahora el boton envía datos al servidor para que sean almacenados. Los datos que el celular envía a MySQL son: IMEI, clave API de *Google (token)*, lo anterior aparece en las Figuras 48, 49, 50 y 51.

---

<sup>8</sup>Estos conceptos y características fueron tratados con anterioridad en las secciones 2.1.4 y 3.4.2

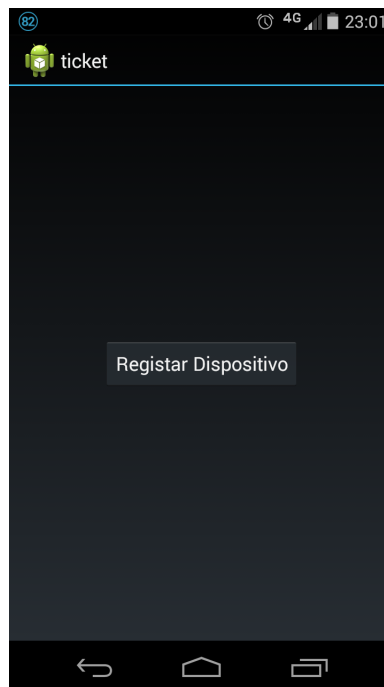


Figura 48. Registrar dispositivo en MySQL.

```
public void registrarDispositivo(String imei, String token, String servicio) {
    JSONMain rD = new JSONMain(this, imei, token, servicio);
    rD.execute();
}
```

Figura 49. Código de función registrar dispositivo.

```
protected JSONObject doInBackground(String... arg0) {
    FuncionesApp FA = new FuncionesApp();
    JSONObject json = FA.registrarDispositivo(imei_, token_, servicio_);
    return json;
}
```

Figura 50. Código clase asíncrona JSONMain.

```
public JSONObject registrarDispositivo(String imei, String token, String servicio){
    String url = "http://192.168.1.100/php/index.php";

    List<NameValuePair> params = new ArrayList<NameValuePair>();
    params.add(new BasicNameValuePair("tag", id_tag));
    params.add(new BasicNameValuePair("imei", imei));
    params.add(new BasicNameValuePair("token", token));
    params.add(new BasicNameValuePair("servicio", servicio));

    //objeto JSON
    JSONObject json = jsonParser.getJSONFromUrl(url, params);
    // return json
    return json;
}
```

Figura 51. Código de función registrarDispositivo() en clase FuncionesApp().

Los datos cuando son enviados con JSON al servidor central llegan al webservice que tiene por nombre: index.php (en la Figura 51, se aprecia la variable 'url' que contiene la ubicación de éste.) y que a través del método POST recibe entre los datos una variable llamada 'tag' que sirve para discriminar que método debe ser utilizado dentro del mismo para realizar alguna tarea específica. Ver Figura 52.

```
// verificamos el tag
if ($tag == 'id')
{
    $imei = $_POST['imei'];
    $token = $_POST['token'];
    $servicio = $_POST['servicio'];
    $turno = $_POST['turno'];

    // verificamos que imei esté registrado
    if ($db->existeImei($imei, $servicio))
    {
        // si existe imei se envia mensaje de error
        $response["error"] = 2;
        $response["error_msg"] = "Ya existe dispositivo";
        echo json_encode($response);
    }
    else
    {
        $query = $db->guardarDispositivo($imei, $token, $servicio);
        if ($query)
        {
            // dispositivo registrado correctamente en la bd
            $response["success"] = 1;
            echo json_encode($response);
        }
        else
        {
            // respuesta error al registrar dispositivo en la bd
            $response["error"] = 1;
            $response["error_msg"] = "Error al guardar dispositivo";
            echo json_encode($response);
        }
    }
}
}
```

Figura 52. Porción de código de servicio web.

En la Figura 52 se muestra que se realiza una instancia de la función 'guardarDispositivo()' y se le pasan como argumentos los datos enviados desde el dispositivo móvil para ser almacenados. Tal método, se encuentra en la clase PHP DB\_Functions.php. Esta clase se encarga de comunicarse directamente con la base de datos MySQL, ya sea para realizar una consulta, *SELECT*, o guardar información, *INSERT*. En la Figura 53 se puede ver la implementación de un *INSERT* con la información recibida.

```
public function guardarDispositivo($imei, $token, $servicio) {  
  
    $resultado = mysql_query("INSERT INTO dispositivos(imei, token, hora, fecha)  
                             VALUES('$imei', '$token', TIME(NOW()), DATE(NOW()))");  
  
    if ($resultado) {  
        return $resultado;  
    } else {  
        return false;  
    }  
}
```

Figura 53. Código de función guardarDispositivos() en clase DB\_Functions.PHP.

Se puede ver en la Figura 54 que los datos han sido almacenados correctamente en la base de datos que se encuentra en el servidor central.

idDispositivo	imei	token	hora	fecha
64	35498405522...	APA91bFFa8C...	17:51:38	2015-01-21

Figura 54. Datos almacenados correctamente en MySQL.

5.3. Solución final

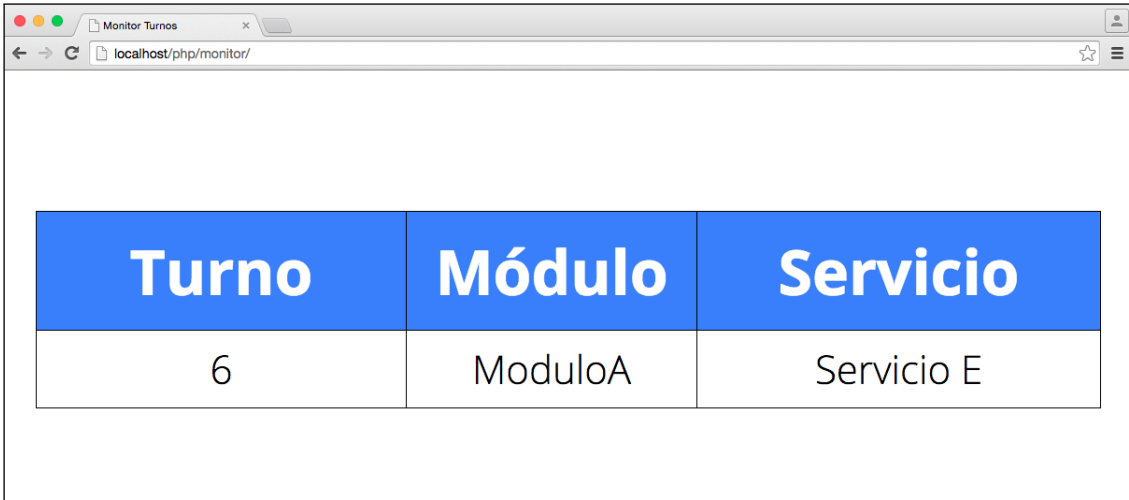
la solución final es el resultado de la cuarta y última iteración del proyecto, de acuerdo al ciclo de vida elegido. Así como fueron descritos anteriormente los prototipos, a continuación en este apartado se describirán las diferentes partes que forman el sistema y sus características.

5.3.1. Servidor central

El servidor central contiene los servicios que actuan como orquestadores del sistema definitivo. Aquí se encuentra la base de datos MySQL, el servicio web (index.php) y la aplicación web (GCM.php).

5.3.2. Monitor

El monitor del sistema tiene por objetivo mostrar los dos últimos números que están siendo atendidos en servicios distintos. Esta pantalla estará ubicada en un televisor ubicado en una zona visible para que los clientes puedan visualizar la información. Se actualiza cuando un número es llamado al módulo de atención. La Figura 55 muestra lo mencionado.



Turno	Módulo	Servicio
6	ModuloA	Servicio E

Figura 55. Monitor del sistema.

5.3.3. Aplicación Web

En la aplicación web es posible seleccionar mediante los menú desplegables el servicio y módulo de atención, ver Figura 56. Lo anterior, sirve para filtrar los mensajes que serán enviados, éstos sólo llegarán a los celulares que tienen un turno de atención en el servicio que permanece seleccionado. Por ejemplo, si está seleccionado el 'Servicio A' y se hace click en el botón 'Siguiente Turno', llegará una notificación a todos los dispositivos que hayan sacado un turno para ese servicio. En la misma figura, abajo es posible mirar el turno actual que está siendo atendido. Existe una barra de menú con diferentes opciones. En 'Reportes' es posible generar reportes con estadísticas relacionadas con el flujo de atención en los diferentes servicios. En 'Monitor' se da la posibilidad que se conozca lo mismo que están viendo los clientes. En la opción 'Configurar', están las opciones para agregar nuevos servicios y módulos de atención.

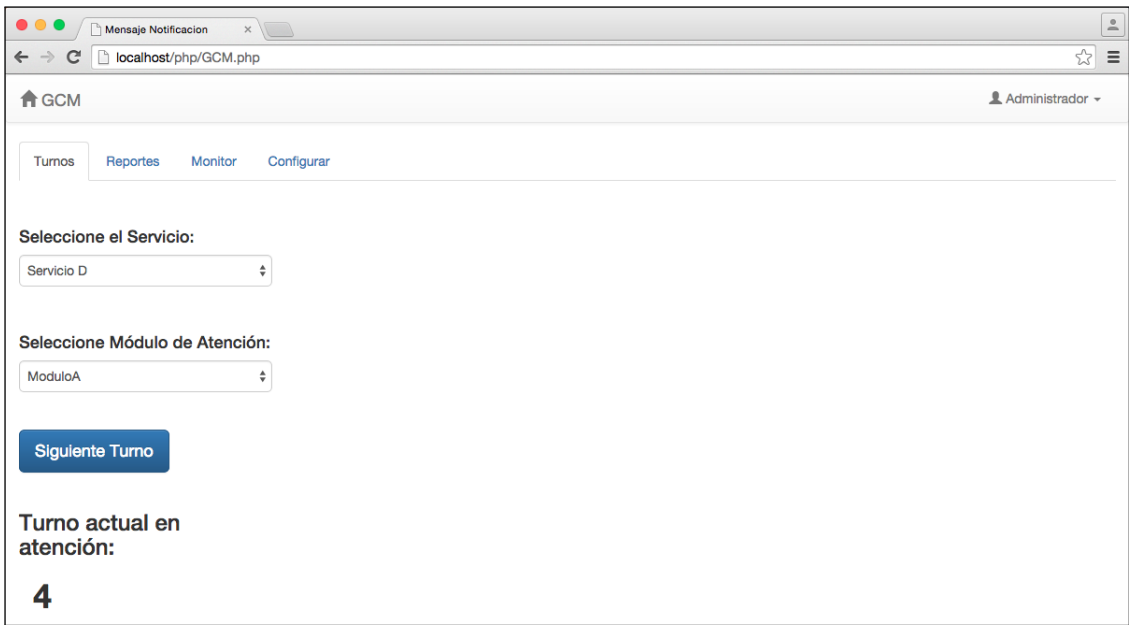


Figura 56. Aplicación web.

### 5.3.4. Aplicación Móvil

En las siguientes Figuras: 57, 58, 59 y 60, se presentan las pantallas que tiene la aplicación móvil. Sus funciones fueron explicadas anteriormente en el Caso de Uso Real CU03 en la sección 4.5.1.

En la Figura de abajo, a la izquierda se aprecia la pantalla principal de la aplicación. Se dispone de botones en donde cada uno corresponde a un tipo de servicio que proporciona el centro de atención. A la derecha, está la misma ventana pero se exhibe el menú que aparece luego de presionar los 'tres puntos verticales' en la esquina superior derecha de la pantalla. Cada opción lleva a una nueva pantalla que contiene:

- **Turno:** Abre la pantalla donde se muestra el turno de atención y el servicio correspondiente donde se obtuvo un ticket.
- **Notificaciones:** Abre una pantalla de la aplicación que muestra el último mensaje enviado desde el servidor central, contiene información del estado de avance de la fila de espera.
- **Información:** Se abre una pantalla que entrega información sobre el proyecto.

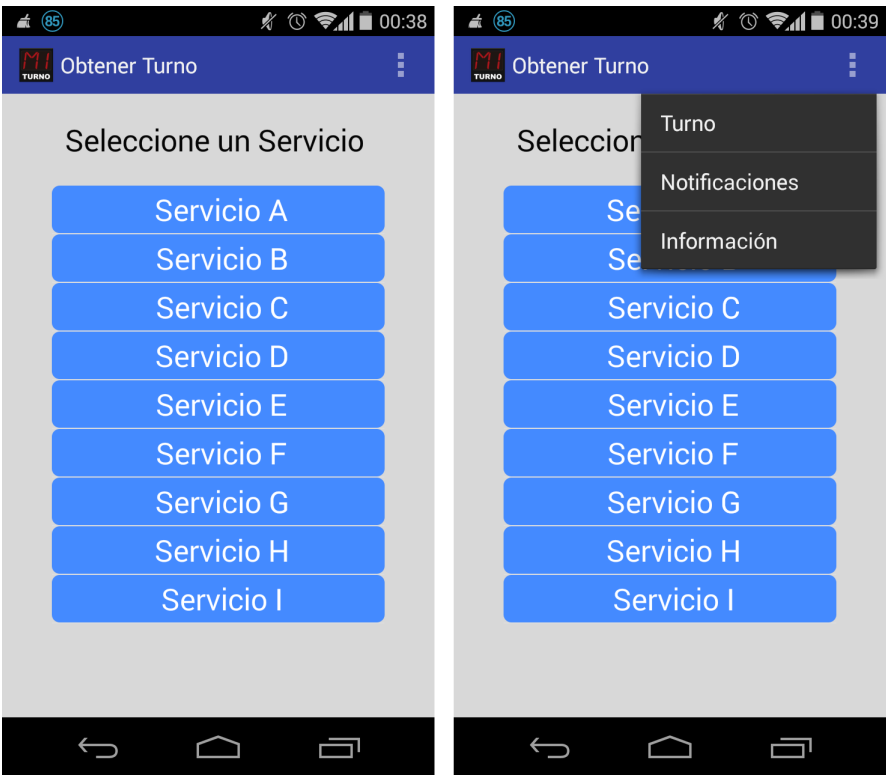


Figura 57. Pantalla principal 'Obtener Turno', a la derecha, y 'Menú' de opciones, a la izquierda,.

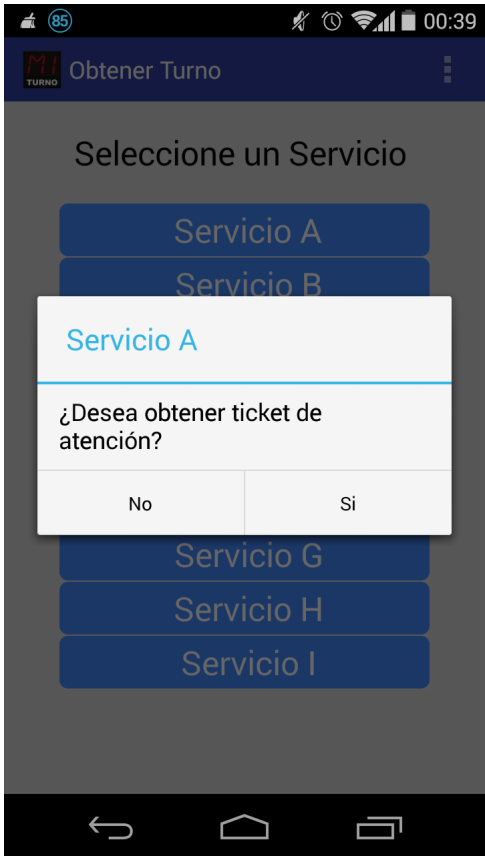


Figura 58. Dialogo confirmación obtener turno.



Figura 59. Pantalla 'Turno Actual'.

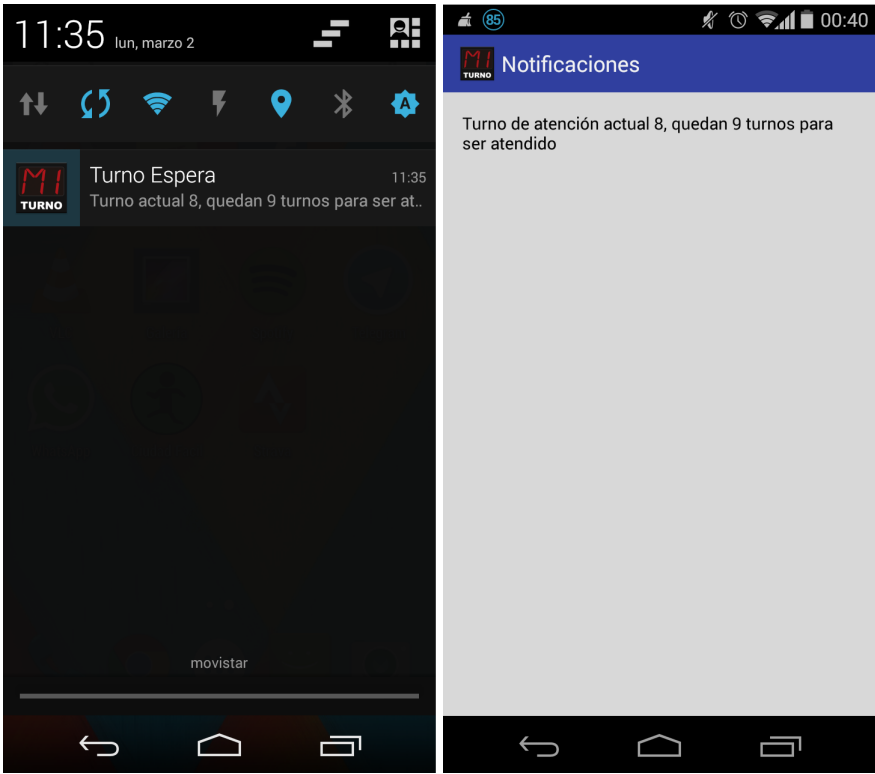


Figura 60. Pantalla con Notificación Push, a la izquierda. Visualización de la Notificación a la derecha.



## 6. PRUEBAS

Las pruebas que serán mostradas en este último capítulo, corresponden a una demostración de funcionamiento del sistema. Se probará la aplicación: web y móvil. En ambos casos que verificará que los requisitos funcionales (RFQ) se cumplan. Para ello, se mostrarán imágenes en las que se indicará el nombre del RFQ que se está validando.

NOTA: Esta demostración está respaldada por un video que será exhibido en la defensa de este proyecto de tesis.

### 6.1. Prueba de Aplicación web.

El orden de las siguientes pruebas obedecen a la Tabla 3. Requisitos funcionales, desde el RFQ 001 - 007.

#### 6.1.1. RFQ - 001 Iniciar sesión en aplicación web.

En la Figura 61 se muestra el inicio de sesión, en los campos: Nombre de usuario y Contraseña se ingresa el usuario y contraseña por defecto ,admin.<sup>en</sup> ambos campos.

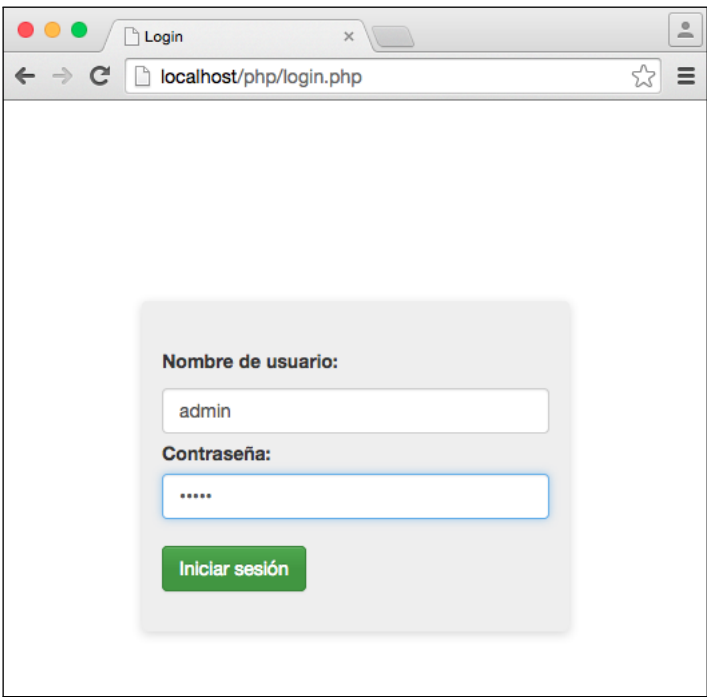


Figura 61. RFQ-001 Ventana Inicio sesión.

Después de presionar el botón: "Iniciar sesión" se abrirá la página principal con acceso a todas las funcionalidades del sistema. Ver Figura 62.

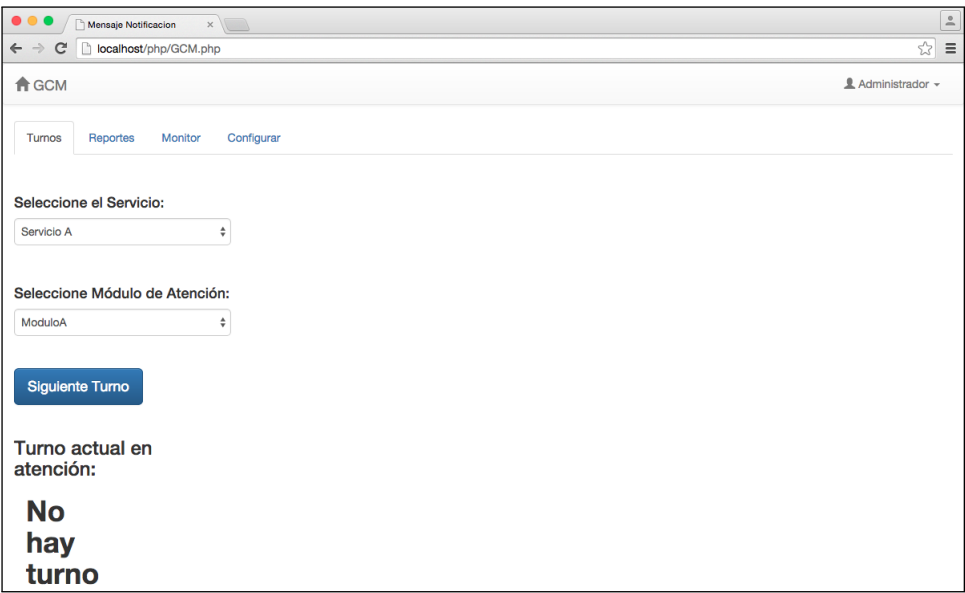


Figura 62. RFQ-001 Ventana principal aplicación web.

**6.1.2. RFQ - 002 Cargar y mostrar servicios.**

El combobox que se aprecia en la Figura 63, contiene los servicios que provee el sistema, estos datos son traídos desde base de datos. Por ejemplo, corresponde a los servicios que entregaría el centro de atención donde se implemente el sistema de gestión de tickets.

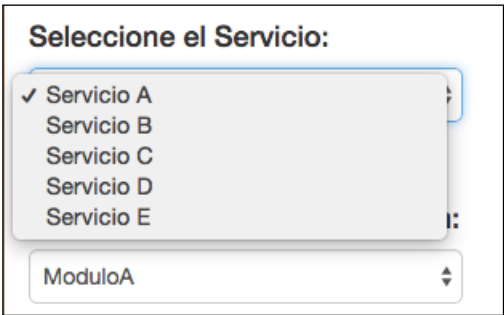


Figura 63. RFQ-002 Combobox con servicios.

**6.1.3. RFQ - 003 Cargar y mostrar módulos de atención.**

Este requisito es similar al anterior. En la Figura 64 se exhiben, los datos cargados desde la base de datos, los módulos de atención que están prestando un servicio en el centro de atención.

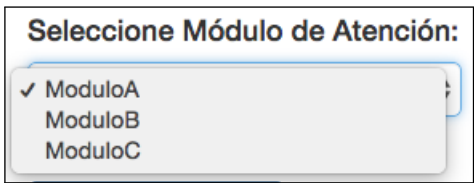


Figura 64. RFQ-003 Combobox con módulos de atención.

**6.1.4. RFQ - 004 Enviar notificaciones desde aplicación web a *smartphone*.**

Este botón, Figura 65, utiliza los dos requisitos anteriores (RFQ-001 y RFQ-002). Se encarga de enviar notificaciones exclusivamente a los dispositivos que han tomado un ticket de atención en el servicio que esté seleccionado en RFQ-001. También se envía el nombre del módulo, que será mostrado en el dispositivo móvil cuando sea el turno del cliente y pueda saber a qué módulo de atención debe pasar para ser atendido.



Figura 65. RFQ-004 Botón para enviar notificaciones.

**6.1.5. RFQ - 005 Generar reportes.**

Los reportes se generan automáticamente al hacer click en la pestaña Reportes". Utilizan los datos almacenados en la base de datos. Se dispone de un gráfico de barras y uno de líneas. El primero, Figura 66, muestra un resumen del número de atenciones que ha tenido cada servicio el último año. El segundo, Figura 67, corresponde al detalle de la cantidad de atenciones que ha tenido cada servicio mensualmente el último año.

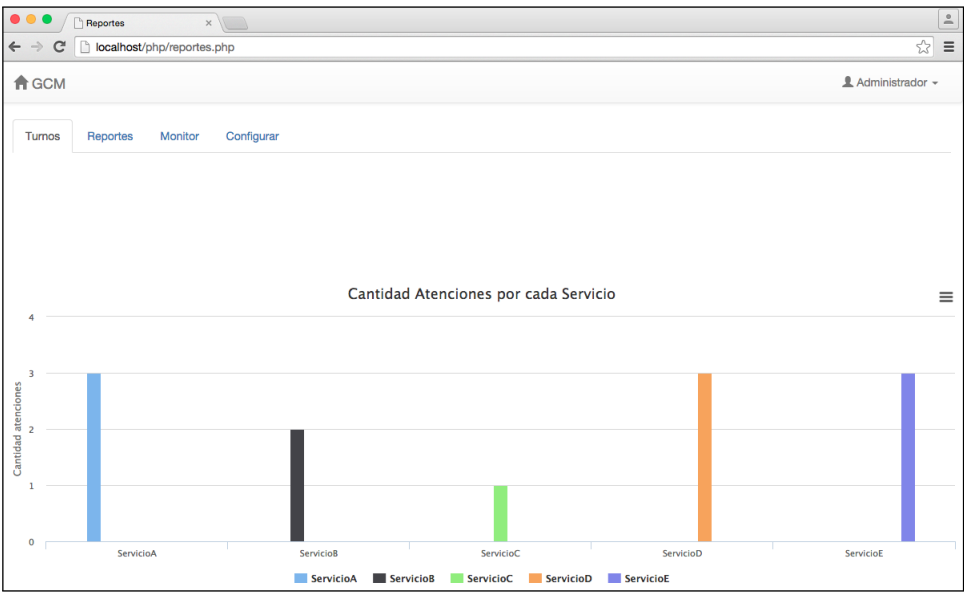


Figura 66. RFQ-004 Gráfico barra número de atenciones por servicio.

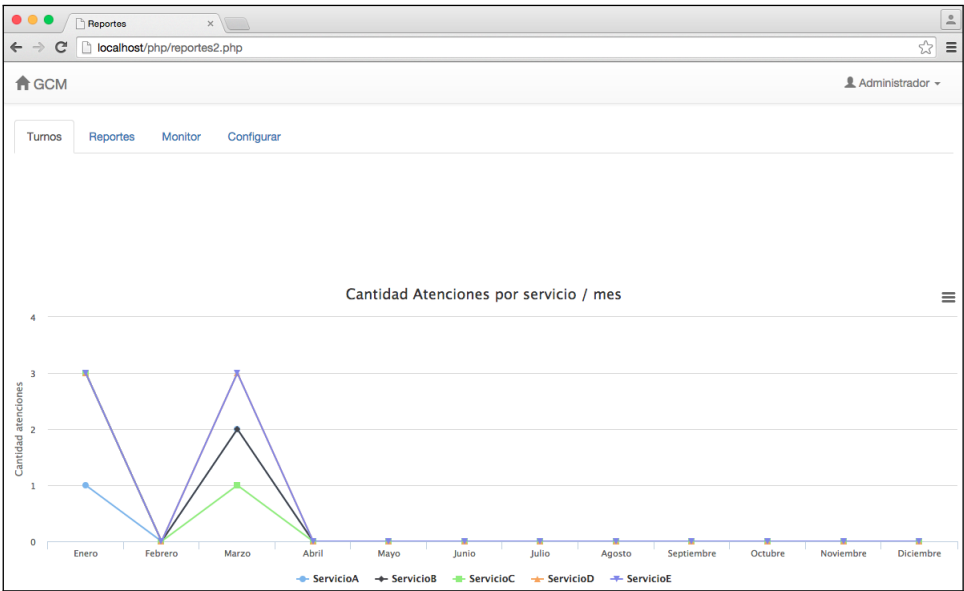


Figura 67. RFQ-004 Gráfico líneas número de atenciones al mes.

6.1.6. RFQ - 006 Mostrar últimos turnos en atención.

Al final de la ventana en la figura 68, se aprecia la etiqueta "Turno actual en atención:". Bajo ésta, se entrega el número de ticket que está siendo atendido correspondiente al servicio que está seleccionado, RFQ-001.

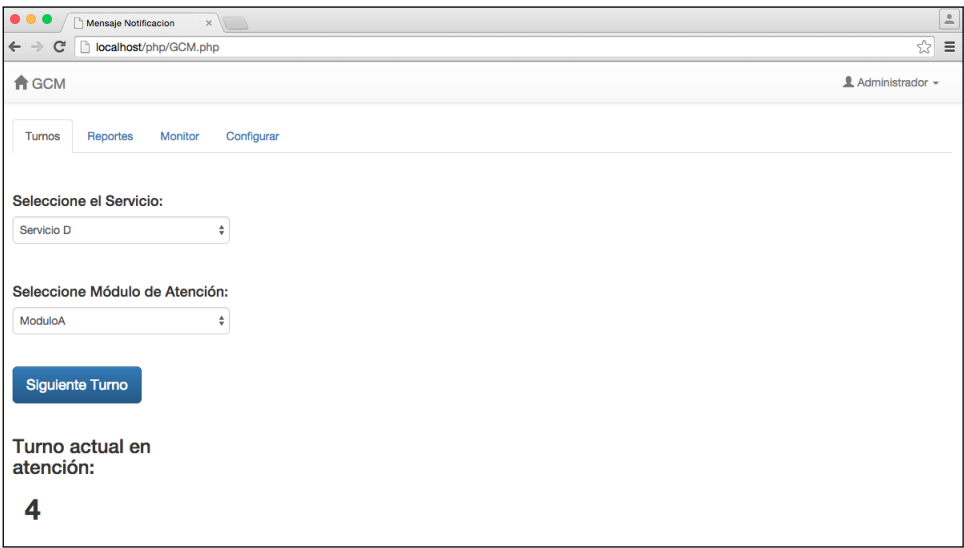


Figura 68. RFQ-006 Turno en atención correspondiente al Servicio D.

**6.1.7. RFQ - 007 Monitor de atenciones.**

El monitor de atenciones, Figura 69, cumple las funciones de informar a los clientes, que están esperando en el centro de atención, a qué módulo de atención deben pasar. para ello, se muestra el número de turno y el tipo de servicio.

Monitor Turnos		
localhost/php/monitor/		
Turno	Módulo	Servicio
6	ModuloA	Servicio E

Figura 69. RFQ-007 Monitor muestra turno, módulo y servicio.

**6.2. Prueba de Aplicación móvil.**

El orden de las siguientes pruebas obedecen a la Tabla 3. Requisitos funcionales, desde el RFQ 008 - 013.

En el ítem anterior, 5.3.4 Aplicación Móvil, ya se explicó lo que muestra la aplicación móvil. La diferencia es que acá se hace énfasis en el cumplimiento de los requisitos funcionales. A continuación se muestra la secuencia normal que surge al utilizar la aplicación móvil para solicitar un ticket de atención, Figuras: 70, 71, 72, 73, 74.

6.2.1. RFQ - 008 Mostrar servicios.

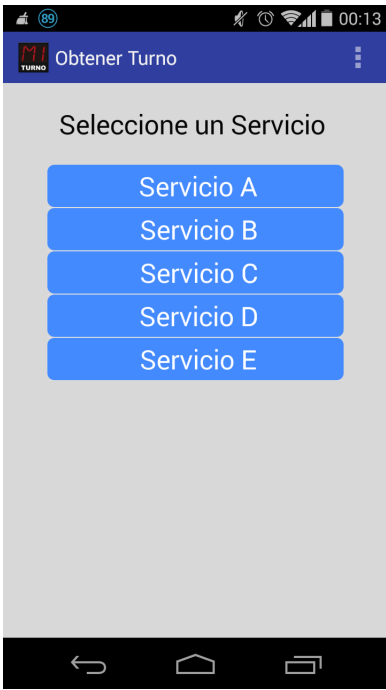


Figura 70. RFQ-008 Servicios disponibles.

6.2.2. RFQ - 009 Seleccionar un servicio.

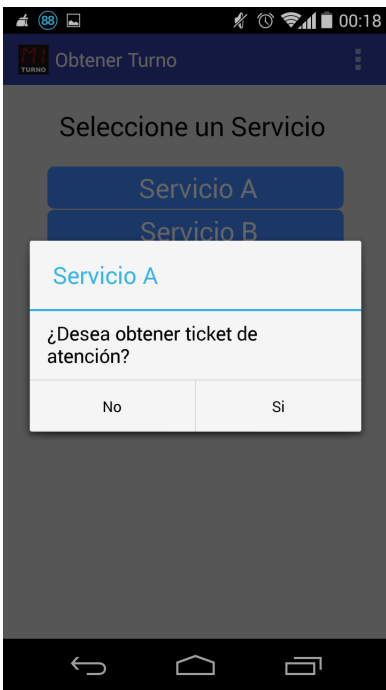


Figura 71. RFQ-009 Selección de Servicio A.

6.2.3. RFQ - 010 Ver turno de atención.

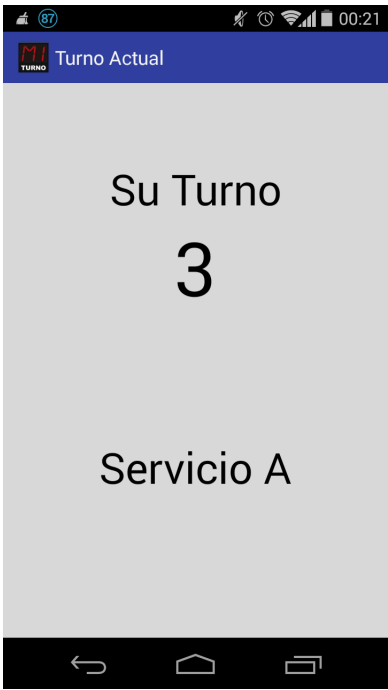


Figura 72. RFQ-010 Turno de atención obtenido.

6.2.4. RFQ - 011 Ver notificaciones.

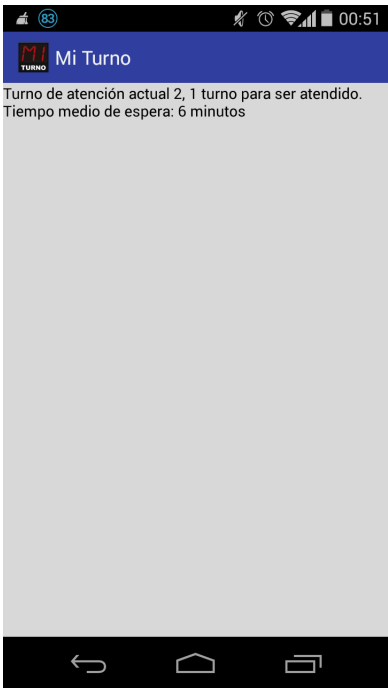


Figura 73. RFQ-011 Notificación recibida desde aplicación web.

6.2.5. RFQ - 012 Mostrar notificaciones nuevas automáticamente.

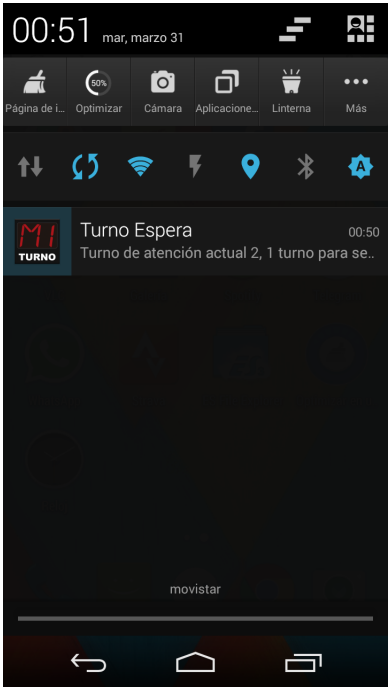


Figura 74. RFQ-012 Notificación Push.

6.2.6. RFQ - 013 Mostrar información de la aplicación.



Figura 75. RFQ-013 Información del proyecto.



## 7. CONCLUSIONES

### 7.1. Conclusiones

El objetivo general de implementar un sistema capaz de solicitar un número de atención a través de una aplicación móvil y realizar el seguimiento de éste utilizando notificaciones push para *Android*, se ha cumplido en su totalidad. Lo anterior queda en evidencia en el capítulo 6.

El análisis previo de las tecnologías disponibles permitió al alumno interiorizarse y elegir adecuadamente las mejores opciones para construir el sistema. Específicamente, en esta etapa es vital responder las preguntas ¿Qué herramientas utilizar? y ¿Cómo comunicar las diferentes piezas de software que formarán el sistema utilizando las tecnologías elegidas?. Resolver dichas interrogantes, ayuda a consolidar una base sólida para desarrollar sin mayores inconvenientes el prototipo y no encontrarse con problemas graves a la mitad de la realización del proyecto. Cometer estos errores provocan desenlaces que significan un desaprovechamiento importante de tiempo y en ocasiones comprometen recursos económicos.

A lo anterior se suma el hecho de lograr correctamente las etapas de análisis y diseño. Para ello, son clave las reuniones iniciales con el cliente o profesor patrocinante. De esta forma, el alumno tiene total claridad de los requisitos del sistema a desarrollar y de los acuerdos tomados para encaminar la solución que se propone a la problemática dada.

La solución que propone este proyecto de titulación presenta una oportunidad real para mejorar la calidad de vida de las personas. De tal forma que los tiempos excesivos de espera, cada día tienden a aumentar y son hechos generadores de desgaste físico y emocional que son capaces de afectar la honra de una persona, sean transformados o disfrazados como una oportunidad para que los clientes aprovechen mejor su tiempo cuando van a realizar una compra o un trámite, pudiendo acudir a otras dependencias y regresar cuando la aplicación le avise que ya se acerca su turno para ser atendido.

Esta solución calza perfectamente en el paradigma de internet de las cosas y también en el contexto de ciudades inteligentes. Día a día se van integrando más artículos cotidianos que antes era impensable que se puedan comunicar con otros dispositivos, o con internet

mismo, incluso no se pensaba en que podían existir. Hoy contamos con la presencia de relojes inteligentes que cuentan con diversas características que desbordan el marco básico que define a un reloj análogo y, que en la medida que su hardware lo permita podemos insertar diversas funcionalidades que estarán disponibles en nuestra muñeca.

## **7.2. Trabajo futuro**

El paso siguiente es incluir esta solución al sistema actual que poseen los centros de atención. Contar con la posibilidad de integrarlo con el sistema rudimentario de los tickets de papel, es una ventaja para cualquier centro pues contaría con un sistema de administración de filas de espera que incluye a todos sus clientes y no discrimina a quienes no tienen un celular inteligente o plan de datos.

Otro punto importante, es proponer un modelo de negocios para este sistema. Este proyecto de título fácilmente puede ser vendido como un servicio central en el que diversas plataformas pueden conectarse a éste y así informar a sus clientes sobre el estado de su turno de atención.

## BIBLIOGRAFÍA

- [And13] Android begin. (2013). Android Cloud Messaging GCM using PHP. Disponible en <http://www.androidbegin.com/tutorial/android-google-cloud-messaging-gcm-tutorial/>. Consultado el 11 de Abril de 2014.
- [And14] Android Developers (2014). Google Cloud Messaging. Disponible en <https://developer.android.com/google/gcm/index.html>. Consultado el 29 de Mayo de 2014.
- [And14] Android Developers (2014). Overview. Disponible en <https://developer.android.com/google/gcm/gcm.html>. Consultado el 09 de Julio de 2014.
- [And12a] Android Hive. (2012). Android Push Notifications using Google Cloud Messaging, PHP and MySQL. Disponible en <http://www.androidbegin.com/tutorial/android-google-cloud-messaging-gcm-tutorial/>. Consultado el 15 de Agosto de 2014.
- [And12b] Android Hive. (2012). How to connect Android with PHP, MySQL. Disponible en <http://www.androidhive.info/2012/05/how-to-connect-android-with-php-mysql/>. Consultado el 16 de Agosto de 2014.
- [And14] Android Developers (2014). Class Overview. Disponible en <https://developer.android.com/reference/com/google/android/gms/gcm/GoogleCloudMessaging.html>. Consultado el 29 de Septiembre de 2014.
- [And12] Android Connect. (2012). Todo sobre las Listviews, ViewHolder y CacheHolder. Disponible en <http://www.androidconnect.org/2012/05/06/todo-sobre-las-listviews-viewholder-y-cacheholder/>. Consultado el 14 de Noviembre de 2014.
- [And] Android curso. (n.d.). La barra de acciones. Disponible en <http://goo.gl/hMw70J>. Consultado el 21 de Noviembre de 2014.
- [And14] Android Developers (2014). Get the Android SDK. Disponible en <http://developer.android.com/sdk/index.html>. Consultado el 09 de Diciembre de 2014.
- [And15] Android Developers (2015). Dashboards. Disponible en <http://developer.android.com/about/dashboards/index.html>. Consultado el 09 de Febrero de 2015.
- [And12] Android eity (2012). Modelo - Vista - Controlador. Disponible en <http://androideity.com/2012/05/10/la-importancia-del-mvc-en-android/>. Consultado el 11 de Febrero de 2015.
- [Ban] Banco de Venezuela. (n.d.). Cola virtual. Disponible en [http://www.bancodevenezuela.com/?bdv=link\\_personas&cod=922](http://www.bancodevenezuela.com/?bdv=link_personas&cod=922). Consultado el 07 de Junio de 2014.
- [Bus14] Business Insider (2014). This Chart Shows Google's Incredible Domination Of The World's Computing Platforms. Disponible en <http://read.bi/1kU8eo0>. Consultado el 08 de Diciembre de 2014.

- [Beg12] Begoña Eguía e Ixone Alonso (2012). El desarrollo de las tecnologías de la información y la comunicación: un nuevo reto para el mercado de trabajo. Disponible en <http://www.ub.edu/geocrit/sn/sn119-74.htm>. Consultado el 27 de Diciembre de 2014.
- [Cus] Customer Attention System. (n.d.). Sistema de Gestión de Colas. Disponible en <http://www.qm-cas.com/index.html>. Consultado el 13 de Abril de 2014.
- [Cis11] Cisco Internet Business Solutions Group. (2011). The Internet of Things. Disponible en [http://www.cisco.com/web/about/ac79/docs/innov/IoT\\_IBSG\\_0411FINAL.pdf](http://www.cisco.com/web/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf). Consultado el 06 de Febrero de 2015.
- [Dev13] DeveloperEconomics (2013). Developer Economics Q3 2013. Disponible en <http://www.developereconomics.com/reports/q3-2013/>. Consultado el 08 de Diciembre de 2014.
- [Elb] El baul de Android (2013). ActionBar y Tabs en Android. Disponible en <http://elbauldeandroid.blogspot.com/2013/10/actionbar-android-en-construccion.html>. Consultado el 21 de Noviembre de 2014.
- [Fay13] Fayerwayer (2013). momit Smart Thermostat, el termostato inteligente para Europa. Disponible en <https://www.fayerwayer.com/2013/12/momit-smart-thermostat/>. Consultado el 02 de Diciembre de 2014.
- [Fay14] Fayerwayer (2014). 'El internet de las cosas', explicado para todos. Disponible en <https://www.fayerwayer.com/2014/06/el-internet-de-las-cosas-explicado-para-todos/>. Consultado el 13 de Enero de 2015.
- [Gre13] . Green Momit. (n.d.). momit Smart Thermostat. Disponible en <http://www.greenmomit.com/thermostat.html>. Consultado el 09 de Enero de 2015.
- [Higa] . High Charts. (n.d.). Column and Bar Chats - Basic column. Disponible en <http://www.highcharts.com/demo/column-basic>. Consultado el 20 de Enero de 2015.
- [Higb] . High Charts. (n.d.). Line Charts - Basic line. Disponible en <http://www.highcharts.com/demo/line-basic>. Consultado el 20 de Enero de 2015.
- [Her2015] . Hermosa Programación. (2015). Parsear datos JSON en Android con JsonReader y Gson. Disponible en <http://www.hermosaprogramacion.com/2015/01/android-json-parsing.html>. Consultado el 25 de Enero de 2015.
- [Ins12] Inside Telecom. (2012). Cola bancaria vía SMS. Disponible en [http://m.insidetele.com/index.php?article\\_id=3713606826767429460](http://m.insidetele.com/index.php?article_id=3713606826767429460). Consultado el 12 de Abril de 2014.
- [Inf15] Infobae (2015). Internet de las cosas. Disponible en <http://www.infobae.com/2015/01/10/1620107-internet-las-cosas-asi-sera-la-vida-cotidiana-el-futuro>. Consultado el 12 de Enero de 2015.

- [Inn14] Innovacion. (2014). 73 % de las conexiones en Chile es vía celulares. Disponible en <http://www.innovacion.gob.cl/2014/03/73-de-las-conexiones-a-internet-en-chile-es-via-celulares/>. Consultado el 13 de Enero de 2015.
- [Inf14] Infobae (2014). Argentinos crearon la primera valija inteligente. Disponible en <http://www.infobae.com/2014/10/20/1603001-argentinos-crearon-la-primera-valija-inteligente>. Consultado el 10 de Febrero de 2015.
- [Iee14] IEEE Internet of Things (2014). About | IEEE Internet of Things. Disponible en <http://iot.ieee.org/about.html>. Consultado el 06 de Junio de 2014.
- [Idm14] Idmsistemas (2014). Q-sige. Disponible en [http://www.idmsistemas.com/es/sige/sige\\_main.php](http://www.idmsistemas.com/es/sige/sige_main.php). Consultado el 07 de Enero de 2015.
- [IBM12] IBM (2012). IBM and city of Lyon, France to create transportation management center of the future. Disponible en <https://www-03.ibm.com/press/us/en/pressrelease/39440.wss>. Consultado el 07 de Febrero de 2015.
- [Jav13] Java Code Geeks. (2013). Android Listview background row style. Disponible en [http://m.insidetele.com/index.php?article\\_id=3713606826767429460](http://m.insidetele.com/index.php?article_id=3713606826767429460). Consultado el 08 de Enero de 2015.
- [JSO] JSON (n.d.). Introducing JSON. Disponible en <http://www.json.org>. Consultado el 08 de Enero de 2015.
- [Kit] Sistemas Point. (n.d.). Sistema para el control de colas virtuales. Disponible en <http://www.sistemasitpoint.com/SolucionColas.asp>. Consultado el 14 de Abril de 2014.
- [Kit] Kits Ozom. (n.d.). Smart Control. Disponible en <http://www.ozom.com/>. Consultado el 09 de Enero de 2015.
- [Myb13] My bring back. (2013). Overview: Interacting with a remote MySQL database in an Android application. Disponible en <http://www.mybringback.com/android-sdk/12924/android-tutorial-using-remote-databases-php-and-mysql-part-1/>. Consultado el 11 de Febrero de 2015.
- [Pri] Prisma Software Gestión. (n.d.). Gestión y control de colas y turnos. Disponible en <http://www.prismasoftwaregestion.com/soluciones/gestion-de-colas-turnos>. Consultado el 13 de abril de 2014.
- [Php] Php. (n.d.). Operadores para Strings. Disponible en <http://php.net/manual/es/language.operators.string.php>. Consultado el 16 de Octubre de 2014.
- [Pew14] Pew Research Center. (2014). Emerging Nations Embrace Internet, Mobile Technology. Disponible en <http://www.pewglobal.org/2014/02/13/emerging-nations-embrace-internet-mobile-technology/>. Consultado el 06 de Diciembre de 2014.
- [Pho13] PhoneArena (2013). Android's Google Play beats App Store with over 1 million apps, now officially largest. Disponible en <http://bit.ly/Jim4kT>. Consultado el 08 de Diciembre de 2014.

- [Pre90] Presidência da República (1990). Ley de consumo. Disponible en [http://www.planalto.gov.br/ccivil\\_03/leis/18078.htm](http://www.planalto.gov.br/ccivil_03/leis/18078.htm). Consultado el 12 de Febrero de 2015.
- [Rfi09] RFID Journal. (2009). That 'Internet of Things' Thing. Disponible en <http://www.rfidjournal.com/articles/view?4986>. Consultado el 06 de Junio de 2014.
- [Sgo12a] Sgoliver. (2012). Notificaciones Push Android. Disponible en <http://www.sgoliver.net/blog/notificaciones-push-android-google-cloud-messaging-gcm-introduccion/>. Consultado el 28 de Abril de 2014.
- [Sgo12b] Sgoliver. (2012). Notificaciones Push Android: Google Cloud Messaging(GCM). Implementación Servidor. Disponible en <http://goo.gl/P47BeR>. Consultado el 28 de Abril de 2014.
- [Sgo13c] Sgoliver. (2013). Notificaciones Push Android: Google Cloud Messaging(GCM). Implementación Cliente. Disponible en <http://goo.gl/ncW8F0>. Consultado el 28 de Abril de 2014.
- [Sgo13d] Sgoliver. (2013). Google Play Services: Introducción y Preparativos. Disponible en <http://goo.gl/2DJ1YR>. Consultado el 02 de Mayo de 2014.
- [Sgo11e] Sgoliver. (2011). Notificaciones en Android (I): Toast. Disponible en <http://www.sgoliver.net/blog/notificaciones-en-android-i-toast/>. Consultado el 03 de Enero de 2015.
- [Sgo11f] Sgoliver. (2011). Notificaciones en Android (III): Diálogos. Disponible en <http://www.sgoliver.net/blog/notificaciones-en-android-iii-dialogos/>. Consultado el 17 de Octubre de 2015.
- [Sgo11g] Sgoliver. (2011). Notificaciones en Android (II): Barra de Estado. Disponible en <http://www.sgoliver.net/blog/notificaciones-en-android-ii-barra-de-estado/>. Consultado el 03 de Enero de 2015.
- [Sch04] Scheihing, E. (2004). Análisis de Desempeño de Sistemas Computacionales, INFO 273. Universidad Austral de Chile, 5 - 6.
- [The14] The Big. (2014). Internet de las cosas: año 2014. Disponible en <http://blogthinkbig.com/internet-de-las-cosas-ano-2014/>. Consultado el 10 de Febrero de 2015.

## **Anexo A: Traducción a español extracto Ley de consumo Brasileña (Ley nº 8.078/90).**

En el caso brasileño, la ley de consumo (Ley nº 8.078/90) define como consumidor y proveedor: .<sup>A</sup>rt. 2º: Consumidor es toda persona física o jurídica que adquiere o utiliza un producto o servicio como destinatario final. Art. 3º: Proveedor es toda persona física o jurídica, pública o privada, nacional o extranjera, así como los entes despersonalizados, que desarrollan actividades de producción, montaje, creación, construcción, transformación, importación, exportación, distribución o comercialización de productos o servicios."

Según la ley de consumo en Brasil, el proveedor debe indemnizar independiente de culpa, es la llamada responsabilidad objetiva del proveedor. Así dispone el artículo 14 de la ley 8.078/90: .<sup>E</sup>l proveedor de servicios responde, independientemente de la existencia de culpa, por la reparación de los daños causados a los consumidores por defectos relativos a la prestación de los servicios, así como por informaciones insuficientes o inadecuadas sobre el uso y riesgos."

Muchos tribunales a lo largo del territorio brasileño han unificado el entendimiento en sus decisiones sobre la reparación por espera en la fila de atención.

Los jueces del Tribunal de Mínima Cuantía del Estado de Paraná, incluso han editado un enunciado que expresa la posición de esta corte sobre el caso: Enunciado N.º 2.7– Fila de banco – daño moral: El tiempo de espera en la fila de una sucursal bancaria, en tiempo excesivo, caracteriza falla en la prestación del servicio y ocasiona reparación por daños morales.

La jurisprudencia de la Corte de Apelaciones del Estado de Sergipe apunta: APELACIÓN CIVIL. ACCIÓN DE INDEMNIZACIÓN POR DAÑOS MORALES ORIGINARIO DE LA ESPERA PARA ATENCIÓN EN FILA DE SUCURSAL BANCARIA. DEMORA INJUSTIFICADA EN LA ATENCIÓN. RESPONSABILIDADE CIVIL CONFIGURADA. DESÍDIA QUE AFRONTA LA DIGNIDADE DE LA PERSONA HUMANA. DANO MORAL SUSCEPTIBLE DE REPARACIÓN PECUNIÁRIA. SÚMULA Nº 04, DO TJSE. La larga espera en fila de sucursal bancaria, además del límite temporal impuesto por la ley municipal, es un hecho generador de desgaste físico y

emocional, capaz de afectar la honra subjetiva de la persona y attingir un derecho inmaterial suyo, originando, por lo tanto, daño moral susceptible de reparación pecuniaria. Reforma de la sentencia, para condenar el demandado en el pago de indemnización en el valor de R\$ 1.000,00 (mil reales), a título de daños no patrimoniales, con interés de 1 % (un por ciento) mensuales, desde la fecha del evento (29/08/2012) y corrección monetaria por el INPC, a partir de esta fecha (Corte de Apelaciones de Sergipe - AC 201300226617; Ac. 678/2014; Segunda Cámara Civil; Rel. Des. Cezário Siqueira Neto; Julg. 17/02/2014; DJSE 11/03/2014).