UNIVERSIDAD CENTRAL DE VENEZUELA
FACULTAD DE CIENCIAS
ESCUELA DE COMPUTACIÓN

DESARROLLO DE UNA APLICACIÓN MÓVIL PARA LA AUTOMATIZACIÓN DE FUERZAS VENTAS

Trabajo Especial de Grado presentado ante la Ilustre
Universidad Central de Venezuela
Por el Bachiller
Alejandro Terreros

Para optar al título de
Licenciado en Computación

Tutor: **Prof. Wilfredo Rangel**

Caracas, Octubre de 2016

ACTA

Quienes suscriben, miembros del Jurado designado por el Consejo de Escuela de Computación, para examinar el Trabajo Especial de Grado presentado por el bachiller **Alejandro Terreros C.I.: 14.121.763**, con el título: **"Desarrollo de una Aplicación Móvil para la Automatización de Fuerza de Ventas"** a los fines de optar al título de Licenciado en Computación, dejan constancia de lo siguiente:

Leído como fue, dicho trabajo por cada uno de los miembros del jurado, se fijó el día 13 de Octubre del 2016 a las 9:00 p.m., para que su autor lo defendiera en forma pública, lo que se hizo en el centro de computación de la Escuela de Computación, mediante una presentación oral de su contenido, luego de lo cual respondió las preguntas formuladas. Finalizada la defensa pública del Trabajo Especial de Grado, el jurado decidió aprobarlo con la nota de 18 puntos.

En fe de lo cual se levanta la presente Acta, en Caracas a los 13 días del mes de Octubre del año 2016.

Prof. Wilfredo Rangel **(Tutor)**

Prof. YoslyHernandez**(Jurado)**

Prof. Jaime Parada**(Jurado)**

Dedicatoria y Agradecimientos

A Dios, a mis padres Santiago Terreros (Q.E.P.D.) y Zara Maurera de Terreros, por apoyarme en todo momento, a mi esposa Johana Mendoza por acompañarme y darme ánimos para concluir esta fase de preparación en mi vida y a mis hijos Andrea y David Terreros por darme esa inspiración extra para seguir adelante, a mis hermanos Santiago y Leonardo Terreros, y al resto de mis familiares por estar presente en todos momentos claves de mi vida.

A mis profesores por ser ejemplo de constancia y perseverancia, por estimular y fomentar el deseo de querer aprender cada día más, por despertar en mí aún más el amor y pasión por la computación, facilitándome herramientas para poder desarrollar las potencialidades que me han permitido ser una persona de éxitos. Gracias por exigirnos tanto y enseñarme que adquirir un conocimiento te fortalece, y te prepara para desempeñar un cargo profesional y que debemos esforzarnos para ser ejemplo.

A mi tutor Wilfredo Rangel, por apoyarme incondicionalmente dándome su valioso tiempo, incentivando y confiando en mí, para llevar a cabo esté trabajo especial de grado.

Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación

DESARROLLO DE UNA APLICACIÓN MÓVIL PARA LA AUTOMATIZACIÓN DE FUERZAS VENTAS

Autor: Br. Alejandro Terreros.
Tutor: Prof. Wilfredo Rangel.
Fecha: 13 de Octubre de 2016.

Resumen

La empresa Distribuidora de Quesos Galicia 300, C.A. encargada de brindar servicio de distribución en el ramo de Alimentos, abarcando una gama de más de quinientos productos (Artículos), atendiendo a través de su fuerza de ventas a más de mil empresas jurídicas (Clientes) distribuidas entre los estados: Miranda, Vargas, Aragua y el Distrito Capital. En primera instancia cada vendedor visita diariamente a un grupo de clientes (ruta), para ofrecer los productos disponibles para su comercialización, lleva un listado impreso, una vez solicitada la mercancía se llena un talonario de manera manual, y al finalizar la ruta diaria son llevados a la empresa, para comenzar allí con el proceso de facturación y despacho de los mismos; en el caso de los vendedores foráneos estos mandan los pedidos por fax o fotos de sus dispositivos móviles, todo esto trae como consecuencia un costo muy elevado entre: impresiones de listas de precios, talonarios, servicio de envió de fax, etc. así como perdida tiempo valioso en el traslado diario a la compañía; el mismo podría reducirse con el uso de herramientas tecnológicas adecuadas que permitan centralizar y visualizar la información en una misma plataforma, convirtiéndola en información útil y organizada, ahorrando tiempo y haciendo la toma de decisiones más eficiente. Bajo el contexto descrito, el presente Trabajo Especial de Grado (T.E.G) se combinación de varios Lenguajes y Herramientas de Programación tales como: Java, Php, Xml, Visual FoxPro y Eclipse, así como varios sistemas manejadores de bases de datos: Mysql, SQLite y SQL Server, para modelar el intercambio de la información contenida entre el Sistema Administrativo Profit 2K8, el cual contiene su data en SQL Server, la Aplicación móvil con SQLite, y el almacenaje de datos en el portal de integración bajo Mysql, mediante la metodología ágil programación extrema.

Palabras Clave: Android, PHP, Programación Extrema, MySql, SQL Server, Visual Foxpro.

Tabla de Contenido

Índice de Figuras

Índice de Tablas

Introducción

En las últimas décadas la internet ha evolucionado enormemente, desde ser inicialmente un mecanismo de entrega de contenidos, hasta convertirse en una herramienta de intercambio dinámico de información, que puede consultarse en cualquier móvil o computador con acceso a ésta. Estola ha convertido en la opción principal para el desarrollo de aplicaciones, para automatizar y optimizar los procesos administrativos de cualquier institución.

El documento de T.E.G. se encuentra agrupado de la siguiente manera:

El capítulo I: define el planteamiento del problema resuelto, los objetivos generales y específicos, el alcance y la justificación e importancia de la investigación.

El capítulo II comprende el marco teórico en el que se presenta la base informativa y conceptual de aquellos elementos sobre Android que nos permitirán el desarrollo del proyecto y de los procesos de desarrollo de software a seguir.

El capítulo III comprende el marco teórico referente a la metodología de desarrollo a seguir para la elaboración de la Aplicación, así como los elementos que permitieron definir cada una de las iteraciones necesarias para cumplir con tal fin.

En el capítulo IV se presenta el marco aplicativo donde se explican y documentan todos los pasos realizados para lograr el desarrollo del sistema desde el análisis inicial, planificación, actividades, pruebas y resultados, hasta llegar a la versión final de la aplicación, siguiendo una adaptación del proceso de desarrollo XP.

Finalmente se presentarán los resultados obtenidos de la investigación, las conclusiones a las que llegamos y las recomendaciones y mejoras para trabajos futuros relacionados.

Capítulo I: Problema de Investigación

El proceso de cotizaciones para el departamento de fuerza de ventas de la empresa Distribuidora de Quesos Galicia 300, C.A. representa un elevado grado de complejidad debido a que la interacción cliente - vendedor - empresa debe ser eficiente y oportuno para brindar al cliente una oportunidad de negocio veraz y dar credibilidad optimizando los tiempos de respuesta.

1.1. Contexto

La empresa Distribuidora de Quesos Galicia 300, C.A., es una empresa del ramo de comercialización y distribución de alimentos, tales como: derivados lácteos, embutidos, víveres y cereales, la empresa fue fundada el 25 de Abril de 2.000, y desde su concepción hasta la actualidad ha incrementado gradualmente su fuerza de ventas partiendo de 2 vendedores hasta 18 en la actualidad.

La fuerza de ventas atiende mensualmente a unos 1.200 clientes distribuidos en 18 zonas, pertenecientes a los estados Vargas, Aragua, Miranda, Guárico y el Distrito Capital.

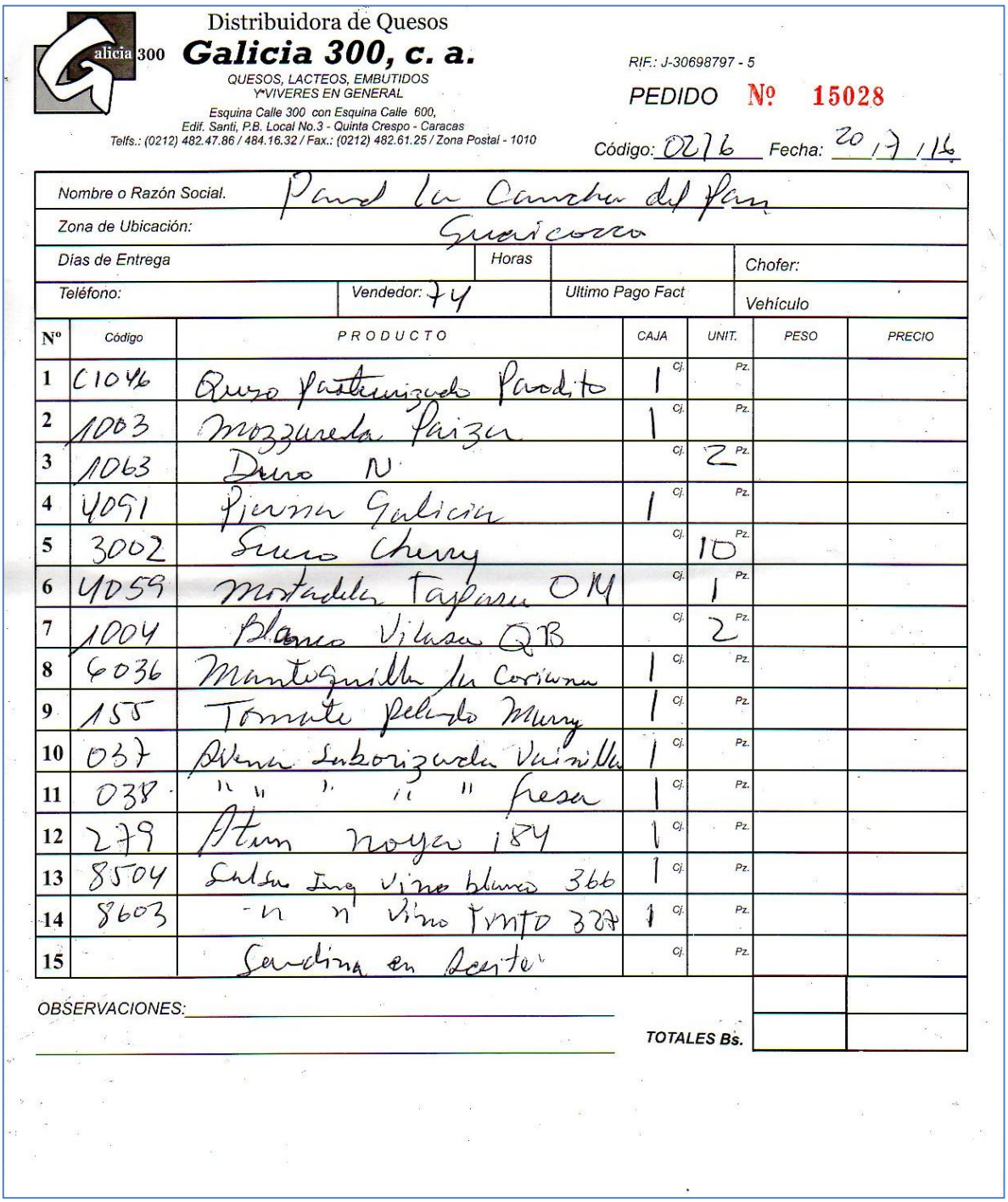
1.1.1. Descripción del Problema

En la distribuidora de alimentos de existe el proceso de carga de pedidos, mediante el cual los vendedores, acuden a sus clientes, visitándolos y tomando sus solicitudes.

Para el momento del diseño del prototipo inicial, el proceso de solicitud de pedidos, era totalmente manual. Esto generaba una gran desorganización en los despachos, ya que debían esperar la recepción de cada uno de los vendedores, podían rondar las 200 solicitudes diarias aproximadamente. Luego de esto, el departamento de crédito y cobranza debía aprobar el pedido recibido.

El proceso como tal, se efectuaba, llenando un talonario de pedidos, que era enviado de manera presencial por los vendedores de Caracas, y vía fax o por foto del teléfono en el

caso de los vendedores de zonas foráneas, para posteriormente ser revisadas por el departamento de crédito y cobranza para su verificación.



Distribuidora de Quesos
Galicia 300, c. a.
QUESOS, LACTEOS, EMBUTIDOS
Y VIVERES EN GENERAL
Esquina Calle 300 con Esquina Calle 600,
Edif. Santi, P.B. Local No.3 - Quinta Crespo - Caracas
Telfs.: (0212) 482.47.86 / 484.16.32 / Fax.: (0212) 482.61.25 / Zona Postal - 1010

RIF.: J-30698797 - 5

PEDIDO Nº 15028

Código: 0276 Fecha: 20 / 7 / 16

Nombre o Razón Social: Pan de la Cancha del Pan
Zona de Ubicación: Guaicoco
Días de Entrega | Horas | Chofer:
Teléfono: | Vendedor: 74 | Ultimo Pago Fact | Vehículo

Nº	Código	PRODUCTO	CAJA	UNIT.	PESO	PRECIO
1	C1046	Queso pasteurizado Paradito	1 Cj.	Pz.		
2	1003	Mozzarella Paizer	1 Cj.	Pz.		
3	1063	Dino N	Cj.	2 Pz.		
4	4091	Pierna Galicia	1 Cj.	Pz.		
5	3002	Suero Cherry	Cj.	10 Pz.		
6	4059	Mortadela Tapara OM	Cj.	1 Pz.		
7	1004	Blanco Vilasa QB	Cj.	2 Pz.		
8	6036	Mantequilla La Coriana	1 Cj.	Pz.		
9	155	Tomate pelado Mary	1 Cj.	Pz.		
10	037	Pierna Enborizada Vainilla	1 Cj.	Pz.		
11	038	" " " " " Fresa	1 Cj.	Pz.		
12	279	Atun Noyer 184	1 Cj.	Pz.		
13	8504	Salsa Ing Vino blanco 366	1 Cj.	Pz.		
14	8603	" " Vino Tinto 378	1 Cj.	Pz.		
15		Sardina en Aceite	Cj.	Pz.		

OBSERVACIONES:

TOTALES Bs.

Figura 1. Formato de Pedidos Manual

Al momento de planificar el desarrollo de una aplicación móvil, la capacidad de sincronizar la información juega un papel muy importante y se puede convertir en una de las tareas más complejas, por lo cual es necesario definir el proceso involucrado para transmitir la información de manera eficiente, para ello se implementa un portal de integración donde se almacenará la información que se encuentra el sistema Administrativo; para su posterior sincronización con el sistema móvil.

Finalmente, se deben desarrollar dos componentes adicionales: uno que realice el trabajo de incluir al sistema Administrativo Profit 2k8, los pedidos enviados en forma de cotización enviados al portal de integración través de la aplicación móvil, y otro que envíe al portal de integración los datos pertinentes para que la aplicación móvil pueda sincronizar la data que necesita la fuerza de venta.

1.1.2. Justificación

Dada la imperiosa necesidad de optimizar los procesos de toma de cotizaciones por parte de la fuerza de ventas y las demoras ocasionadas en la distribución consecuencia de los procesos manuales, lentos y tediosos, que no permitían realizar un despacho de manera oportuna y eficiente se desarrolló una aplicación móvil usable, económica y eficiente ya que acorta el tiempo de distribución, mejorando la capacidad de atención a más clientes.

1.2. Objetivos

1.3.1. Objetivo General

Desarrollar una aplicación móvil para la automatización fuerzas de ventas.

1.3.2. Objetivos Específicos

- Definir la arquitectura que permita desarrollar la aplicación móvil.
- Diseñar la Interfaz de las aplicaciones que permitan dar operatividad y eficiencia a la fuerza de ventas.
- Definir la base de datos que permita la interacción con la aplicación móvil.
- Aplicar pruebas en el sistema móvil.
- Desarrollar módulos de emisión y recepción de pedidos al sistema Administrativo Profit 2K8.
- Desarrollar un reporte para el usuario en el sistema Administrativo Profit 2k8.

1.3. Metodología de desarrollo

Los métodos de desarrollo ágil permiten guiar el proceso de construcción de software y entregar resultados parciales en cada iteración hasta completar el producto esperado. Para el desarrollo de la aplicación móvil esperada se ha seleccionado el método Programación Extrema (XP), adaptado en algunos aspectos para su desarrollo.

1.4. Alcance

El sistema es una herramienta escalable y usable, que se desarrolló para gestionar y agilizar procesos relacionados con las solicitudes de pedidos para los clientes de la empresa Distribuidora de Quesos Galicia 300, C.A.

El sistema tiene la capacidad de sincronizar la data del programa Administrativo Profit 2k8 almacenada en el SMBD SQL Server, y enviarla al portal de integración, tomando para ello únicamente artículos, clientes y cuentas por cobrar de los mismos, filtrando información relevante para la empresa tales como stock, o algún campo especial, exclusiones de vendedores y tipos de precios.

La aplicación Móvil permitirá al usuario consultar artículos con su descripción, precio y stock, así como sus clientes con sus respectivas cuentas por cobrar, así como realizar los pedidos, permitiendo a enviar el envío comentarlo. Sin tomar otros aspectos como realizar la venta o cobranza al cliente.

La aplicación móvil cuenta con una base de datos interna SQLite utilizada para mantener un histórico y poder retransmitir el pedido en caso haber fallado por motivos de comunicación, más no se desarrolló para este TEG informes estadísticos de gestión, tales como pedidos enviados vs procesados, o pedidos sugeridos por clientes.

A nivel del sistema administrativo Profit 2k8 solamente se desarrolló un reporte de pedidos en Android permitiendo al usuario seleccionar los pedidos por rango de código de pedido, código de vendedor y por fecha. Dando como resultado el pedido en formato digital disponible para ser importado al momento de su facturación como se muestra en la Figura 2.

Distribuidora de Quesos

Galicia 300, C.A.

Calle 300 con 600, Edif. Santi Local 3
Quinta Crespo - Telf. 482.4786 /481.4349

Cliente: 2047 PANADERIA LOS CORTIJOS, C.A.
R.I.F.: J-00084215-9 N.I.T.:
Dirección: CALLE B, LOS RUICES

Teléfonos: 234.61.93 FAX:
Placa: _____
Transporte: SUPER3 NESTOR DAVID SANCHES

Pedido: 30007375
Página: 1
Fecha Emisión: 20/07/2016

Condic. Pago: CREDITO 15
Vendedor: 02
GRACIELA GIL (A)

	Código	Descripción	Caja	Unidad	Kgrs	Peso	Precio
1	370	ATUN REY DEL GOLFO NATURAL 24x140	2				1,160.00
2	368	SARDINA REINA DEL CARIBE ITALIANA 48x170	1				375.00
3	365	SARDINA REINA DEL CARIBE AC. 48x170 G.	1				307.00
4	4057	PASTA DE HIGADO Oscar Mayer 12 x 226 Gr	1				670.00
5	4060	CHORIFRITO OSCAR MAYER 8X400 GRS.	1				1,450.00
6	4159	COPPA TIPO PARMA GIACOMELO 15x1.20		4			12,194.00
7	037	AVENA SABORIZADA VAINILLA 12X5X40	2				980.00
8	028	AVENA AVELINA INSTANTANEA 12X400 (VERDE)	5				1,335.00
9	027	AVENA AVELINA TRADICIONAL 12X400 (AMAR)	5				1,165.00
10	M102	MANTEQUILLA PAISA S/SAL 20x200 GRS.	2				1,460.00
11	4205-1	SALAMI TIPO GERONA LA LEONESA	1				8,233.00
12	P2066	QUESO PIRINEO LOS FRAILES BARRA 5x2.7	2				5,500.00
13	239	MARGARINA IND. SIN SAL ASTOR 15 KGS.	5				69,998.50

Figura 2. Formato de Pedido Móvil en Profit 2k8

Capítulo II: Marco conceptual

A partir del año 2.008 los teléfonos móviles han experimentado una gran evolución, desde la comercialización de la primera generación de móvil, El DynaTAC 8000X de Motorola es presentado oficialmente en 1984, año en que se empezó a comercializar. El teléfono pesaba cerca de 1 kg, tenía un tamaño de 33.02 x 4,445 x 8,89 centímetros y su batería duraba una hora de comunicación o una jornada laboral (ocho horas) en espera, con pantalla led, grandes y pesados, pensados sólo para hablar por teléfono en cualquier parte, a los últimos modelos, La generación 4, o 4G, es la evolución tecnológica que ofrece al usuario de telefonía móvil, internet con más rapidez un mayor ancho de banda que permite, entre muchas otras cosas, la recepción de televisión en alta definición.

Es así como nace Android. Android es un sistema operativo y una plataforma software, basado en Linux para teléfonos móviles. Además, también usan este sistema operativo, tablets, notebooks, relojes, reproductores de música, etc. [1]

Android permite programar en un entorno de trabajo (framework) de Java, aplicaciones sobre una máquina virtual Dalvik (una variación de la máquina de Java con compilación en tiempo de ejecución).

Android: Definiciones Básicas y Desarrollo de Aplicaciones.

Fue desarrollado por Android Inc., empresa que en 2005 fue comprada por Google, aunque no fue hasta 2008 cuando se popularizó, gracias a la unión al proyecto de Open Handset Alliance, un consorcio formado por 48 empresas de desarrollo hardware, software y telecomunicaciones, que decidieron promocionar el software libre. Pero ha sido Google quien ha publicado la mayor parte del código fuente del sistema operativo, gracias al software Apache, que es una fundación que da soporte a proyectos software de código abierto.

Dado que Android está basado en el núcleo de Linux, tiene acceso a sus recursos, pudiendo gestionarlo, gracias a que se encuentra en una capa por encima del Kernel, accediendo así a recursos como los controladores de pantalla, cámara, memoria flash, etc.

2.1.Versiones (API Levels)

El historial de versiones del sistema operativo Android se inició con el lanzamiento de Android beta en noviembre de 2007. La primera versión comercial (de prueba), Android 1.0, fue lanzada en septiembre de 2008. Android es un sistema operativo móvil desarrollado por Google y la Open Handset Alliance, y ha visto un número de actualizaciones a su sistema operativo base desde su lanzamiento original. [2]

Estas actualizaciones típicamente corrigen fallos de programa y agregan nuevas funcionalidades. Desde abril de 2009, las versiones de Android han sido desarrolladas bajo un nombre en clave y sus nombres siguen un orden alfabético: Apple Pie, Banana Bread, Cupcake, Donut, Éclair, Froyo, Gingerbread, Honeycomb, Ice Cream Sandwich, Jelly Bean, KitKat, Lollipop y Marshmallow. La actualización más reciente es Android 6.0 Marshmallow, que fue anunciado oficialmente en mayo de 2015.

Debido al avance en la capacidad de procesamiento de dispositivos móviles (y más específicamente en el caso de los teléfonos inteligentes), a partir de la línea 4.xx unifica su uso para cualquier dispositivo. Véase La Figura 3. Distribución de Android según su versión. Información desde la versión 2.2 hasta la versión 6.0 para Mayo 2016. [3]

Version	Codename	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	2.2%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	2.0%
4.1.x	Jelly Bean	16	7.2%
4.2.x		17	10.0%
4.3		18	2.9%
4.4	KitKat	19	32.5%
5.0	Lollipop	21	16.2%
5.1		22	19.4%
6.0	Marshmallow	23	7.5%

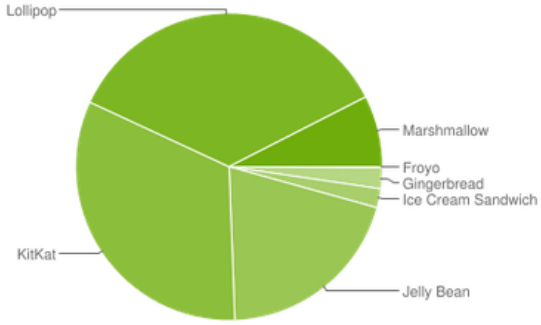


Figura 3. Distribución de Android según su versión. Información desde la versión 2.2 hasta la versión 6.0 para Mayo 2016 https://developer.android.com/about/dashboards/index.html.

2.1.1. Incidencia en el mercado

En julio de 2.016 Android es la plataforma que lidera el *market share* (cantidad de dispositivos con Android), por encima de iOS, Windows Mobile y otros como Symbian OS y RIM OS. Durante los meses de Abril, Mayo y Junio, aunque pudimos ver cierto crecimiento de iOS en el Japón, Reino Unido, Alemania, Francia, Italia, España y Estados Unidos, iOS ha conseguido hacerse con el 35,1% de las ventas de dispositivos inteligentes con Android. En cuanto a China, vemos como Android se ha conseguido hacer con el 85% de las ventas de teléfonos inteligentes. Mientras que Windows ha registrado las mayores pérdidas, Android también ha aumentado la cuota en estos países

Smartphone OS Sales Share (%)

Germany	3 me Jul 15	3 me Jul 16	% pt. Change
Android	73.7	79.8	6.1
iOS	15.2	15.2	0.0
Windows	10.1	4.8	-5.3
Other	1	0.1	-0.9

GB	3 me Jul 15	3 me Jul 16	% pt. Change
Android	54.4	57.3	2.9
iOS	32.8	38.0	5.2
Windows	11.9	4.3	-7.6
Other	1	0.4	-0.6

France	3 me Jul 15	3 me Jul 16	% pt. Change
Android	69.6	75.6	6.0
iOS	17	18.8	1.8
Windows	12.6	4.9	-7.7
Other	0.7	0.7	0.0

Italy	3 me Jul 15	3 me Jul 16	% pt. Change
Android	72.9	82.5	9.6
iOS	11.4	12.7	1.3
Windows	14.3	4.7	-9.6
Other	1.4	0.1	-1.3

Spain	3 me Jul 15	3 me Jul 16	% pt. Change
Android	89.4	90	0.6
iOS	6.6	9.2	2.6
Windows	4	0.6	-3.4
Other	0	0.1	0.1

USA	3 me Jul 15	3 me Jul 16	% pt. Change
Android	65.6	65	-0.6
iOS	30.1	31.3	1.2
Windows	3.8	2.4	-1.4
Other	0.5	1.3	0.8

China	3 me Jul 15	3 me Jul 16	% pt. Change
Android	79.4	85.0	5.6
iOS	18.7	14.3	-4.4
Windows	1.5	0.2	-1.3
Other	0.4	0.5	0.1

Australia	3 me Jul 15	3 me Jul 16	% pt. Change
Android	56.1	60.2	4.1
iOS	34.9	35.2	0.3
Windows	6.9	2.9	-4.0
Other	2.1	1.7	-0.4

Japan	3 me Jul 15	3 me Jul 16	% pt. Change
Android	62.9	64.6	1.7
iOS	35.1	34.1	-1.0
Windows	0.1	0.6	0.5
Other	1.8	0.6	-1.2

EU5	3 me Jul 15	3 me Jul 16	% pt. Change
Android	71.0	77.0	6.1
iOS	17.2	18.4	1.2
Windows	10.9	4.2	-6.7
Other	0.9	0.3	-0.6

Figura 4. Evolución de ventas de dispositivos según su Sistema Operativo, según http://es.kantar.com/tech/m%C3%B3vil/2016/septiembre-2016-cuota-de-mercado-de-smartphones-en-espa%C3%B1a/

2.2. *Arquitectura*

Como sistema operativo, Android está organizado en capas y bibliotecas con diferentes funcionalidades, servicios e interacciones con las demás partes o subsistemas del sistema operativo.

2.2.1. Dalvik Virtual Machine (DVM)

El kernel de Android se encuentra basado en el de Linux, con librerías escritas en C, y aplicaciones corriendo sobre un framework de soporte para la máquina virtual Dalvik (DVM: Dalvik Virtual Machine) con compilación *Just-In-Time* (JIT), con traducción de *bytecodes* de Java (JIT a partir de Android 2.2).[5]

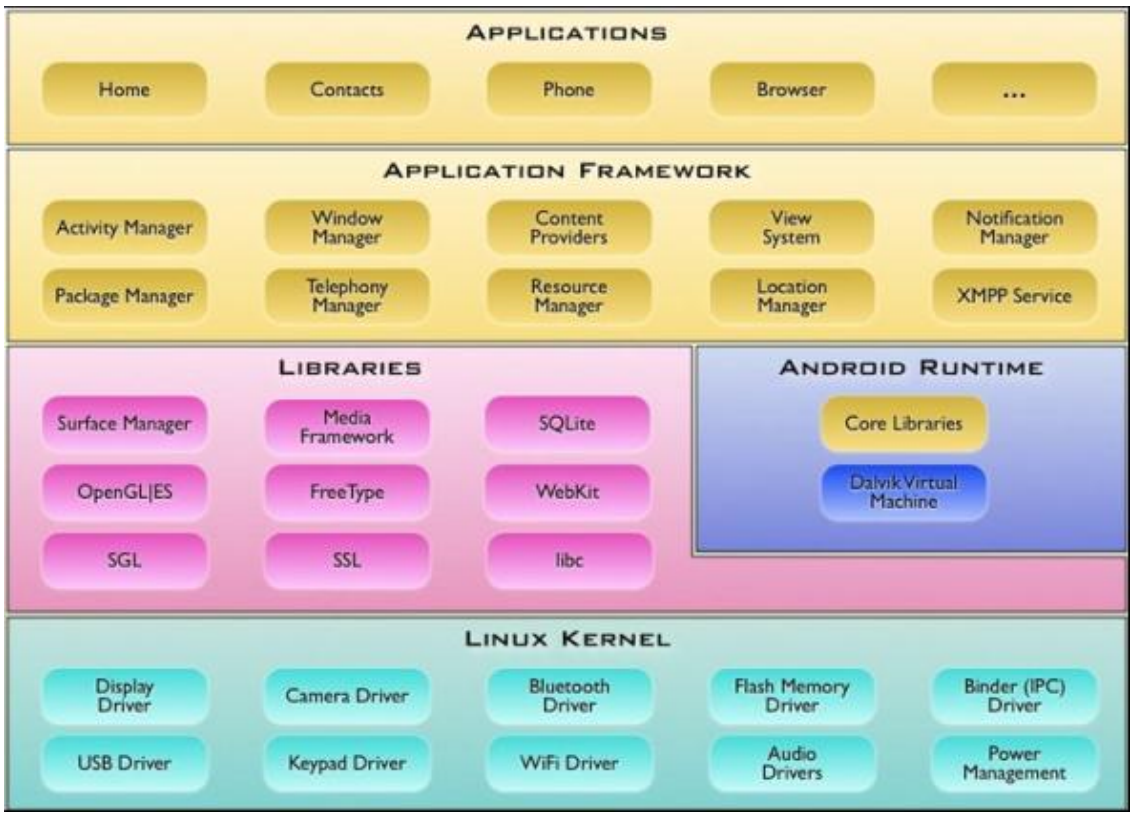


Figura 5. Arquitectura de componentes que conforman Android.
http://developer.android.com/guide/basics/what-is-android.html

Figura 5. Arquitectura de componentes que conforman Android. Muestra la organización en capas de Android. La capa de más bajo nivel o más cercana al hardware es la basada

en Linux y directamente sobre ella se podría decir que se ubica la capa de “traducción” a lo que es propio de Android, donde se pueden identificar dos partes en esta capa:

- Bibliotecas: algunas de las cuales, quizás identificables como las clásicas de Linux, tal como libc y otras más *propias* o *adaptadas para* Android, como por ejemplo *MediaFramework, SQLite*.
- Android Runtime: es quizás la parte de esta segunda capa más específica de Android y diferenciada de lo que estaría a este nivel en un sistema operativo Linux, que contiene las bibliotecas más importantes y la DVM.

2.2.2. Diferencias entre JVM Vs DVM

Aunque desde el punto de vista de la aplicación no hay muchas diferencias entre la JVM y la DVM, arquitecturalmente ambas difieren de manera considerable. Sintéticamente, el código fuente Java es compilado a un archivo .class. Posteriormente, dicho archivo es procesado por *dexer* (una herramienta que es parte de la SDK de Android), a fin de generar el archivo con formato DEX que contiene los bytecodes Dalvik. Esto significa que la DVM sería similar a la JVM en términos prácticos (o *actuaría como tal*), gracias a la compilación *Just-In-Time* de byte codes de Java a código ejecutable Dalvik o dex-code (*Dalvik Executable*). Ver Tabla 1 Diferencias entre DVM y JVM. [6]

Más propiamente, DVM es una máquina virtual de proceso tal como la JVM, con el agregado de tener un JIT para archivos de clase (.class) de Java o producidos por un compilador Java para una JVM estándar.

	DVM	**JVM**
Arquitectura	Registros	Pila
Soporte S.O.	Android	Múltiples
Clases	Dex	Class
Pool de Constantes	Por Aplicación	Por Clase
Aplicación	APK	JAR

Tabla 1 Diferencias entre DVM y JVM.

2.3. Desarrollo de Aplicaciones para Android en Java

2.3.1. Aplicaciones Nativas

Entre las principales ventajas y desventajas del desarrollo nativo en dispositivos móviles basados en Android son los siguientes: [7]

Ventajas:

- Se puede instalar en casi todo tipo de dispositivos, lo cual permite que los diferentes fabricantes y operadoras tomen a este sistema operativo en sus equipos.
- Android tiene un sistema completamente libre para que un desarrollador no solo pueda modificarlos sino también mejorarlo.
- Entorno de desarrollo libre basado en Eclipse y compatible con varios sistemas operativos, que cuenta con simulador de Android para probar las aplicaciones.
- Librerías y componentes reutilizables para todo tipo de aplicaciones.
- Gran comunidad de desarrolladores, varias fuentes de información acerca de la plataforma con ejemplos de cada una de las aplicaciones.
- Más de 100.000 aplicaciones disponibles para teléfonos Android la mayoría gratis.
- Es un sistema operativo multitarea que es capaz de gestionar varias aplicaciones abiertas a la vez, dejando en suspensión aquellas que no se utilicen y cerrarlas en caso de resultar ya inútiles para evitar un consumo de memoria.
- Compatibilidad con base de datos a través de SQLite, una base de datos SQL sencilla de utilizar.

Desventajas:

- Google tiene una política restrictiva hacia las versiones más recientes, no haciéndolas públicas hasta que ellos lo vean conveniente.
- Como Android es un sistema operativo de código abierto produce que este expuesto a vulnerabilidades.

2.3.2. Instalación y configuración de JDK 8

Para instalar el entorno de desarrollo JDK 8 en Windows, debemos seguir estos pasos:

1. Descargar el JDK de la sitio web de Oracle

   http://www.oracle.com/technetwork/java/javase/downloads/index.html, en esta página encontramos las diversas versiones (durante la redacción de estas líneas, la última versión que existe es la versión 8 como se muestra en la Figura 5).

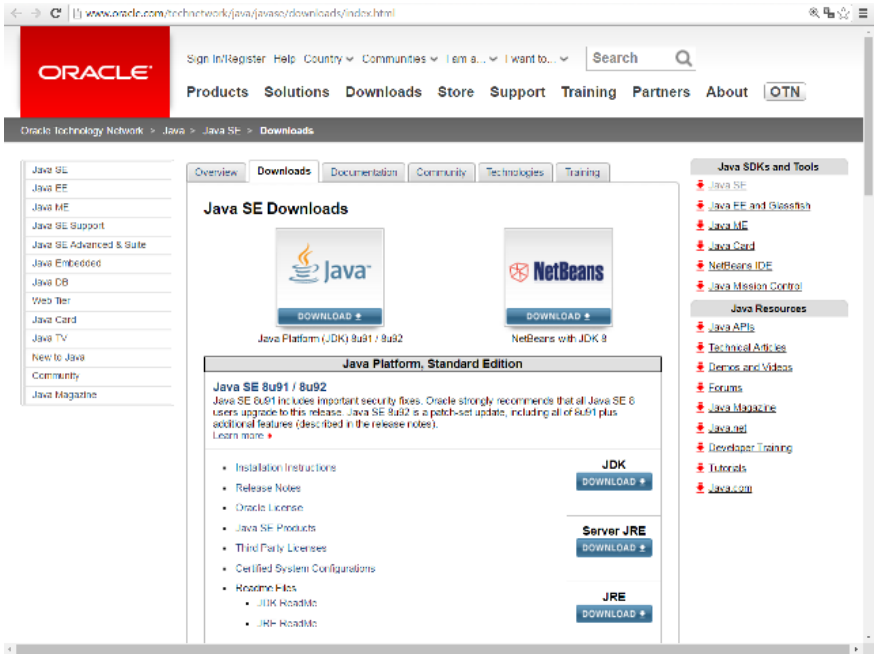


Figura 6. Sitio web de Oracle para la descarga del JDK
http://www.oracle.com/technetwork/java/javase/downloads/index.html.

2. Se procede a aceptar los términos de licencia y escoger el instalador del sistema operativo en el que se está desarrollando, en este caso se escoge Windows.
3. Con esto se descargará un archivo ejecutable, en el cual procedemos a dar doble clic para la instalación y posteriormente damos clic en siguiente.
4. Escogemos las características que deseamos instalar y damos clic en siguiente, se procederá a instalar del JDK.
5. Por último se nos presenta un mensaje que nos informa que la instalación se realizó con éxito, damos clic en cerrar.

2.3.3. Instalación y configuración del SDK de Android

Para instalar el entorno de desarrollo JDK 8 en Windows, debemos seguir estos pasos: [1]

1. Primeramente se descargara la versión de SDK de Android del sitio web http://developer.android.com/sdk/index.html. Véase Figura 7.Sitio web para la descarga del SDK de Android.

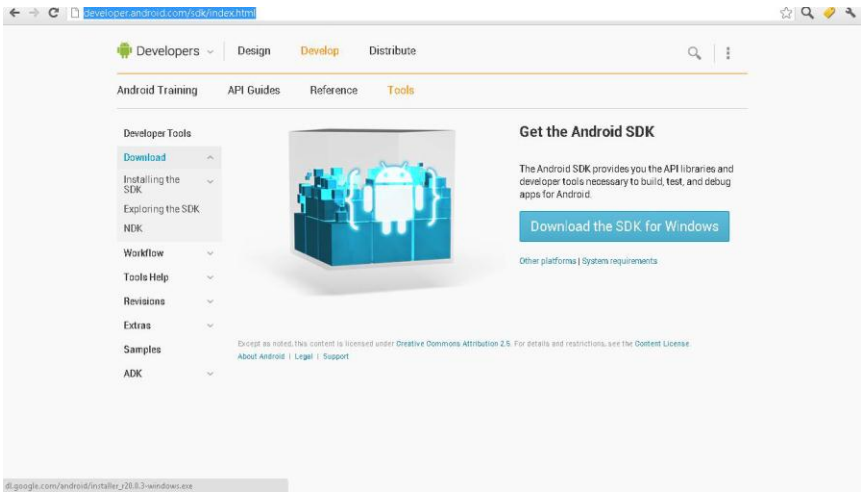


Figura 7.Sitio web para la descarga del SDK de Android

2. Se procede a la instalación del SDK, dando doble clic en el archivo descargado, se nos presenta la siguiente pantalla y damos clic en siguiente.
3. Escogemos el directorio en donde queremos que se instale el SDK y procedemos a dar clic en siguiente.
4. Se instalara todos los paquetes del SDK de Android, damos clic en instalar.
5. Una vez completada la instalación procedemos a dar clic en Finalizar.
6. Posteriormente se deben instalar paquetes adicionales del sistema operativo Android, con esto podemos ejecutar nuestras aplicaciones dentro de un dispositivo virtual, escogemos la última versión de Android, la cual al momento de la redacción de este documento es la 6.0 y se escogerá la versión 2.3.3 (API 10) como base, para realizar las pruebas en las diferentes versiones, damos clic en instalar paquetes.

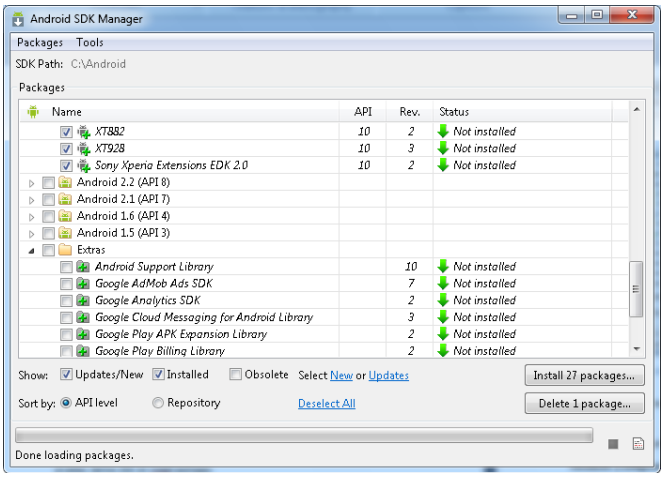


Figura 8. Instalación de las plataformas disponibles de Android

7. Una vez terminada la instalación de los paquetes se procederá a crear una máquina virtual de Android, para esto nos dirigimos al Menú *Tools* y seleccionamos *Manager ADVS*, aquí podemos crear los dispositivos virtuales para la ejecución de las aplicaciones en un entorno virtual.
8. Para crear un nuevo dispositivo virtual damos clic en Nuevo.
9. Ingresamos el nombre del dispositivo virtual, la plataforma en la que se va a ejecutar, asignamos el tamaño de memoria, el tipo de pantalla y el hardware con el que cuenta el dispositivo. Una vez terminado damos clic en *Create ADV*.

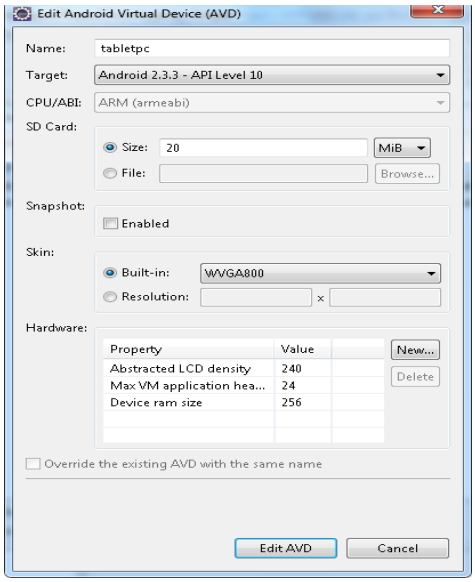


Figura 9. Configuración de un nuevo dispositivo virtual de Android

10. Una vez terminada la creación del dispositivo virtual, podemos iniciarlo para comprobar su correcto funcionamiento, para esto damos clic en comenzar.

11. En la Figura 10 se muestra la ejecución del dispositivo virtual.

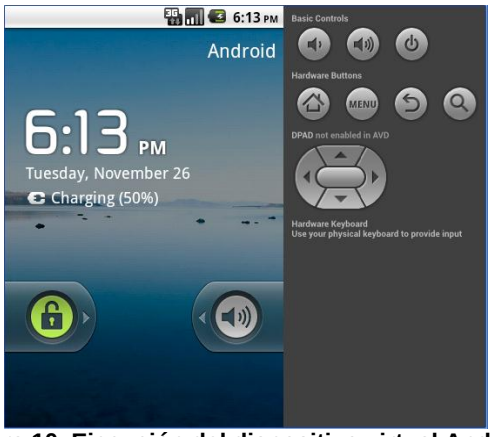


Figura 10. Ejecución del dispositivo virtual Android

2.3.4. Instalación y configuración de Eclipse Kepler 4.3.1

Se procederá a detallar la instalación y configuración del entorno de desarrollo eclipse:

1. Se utilizará Eclipse Kepler 4.3.1 como entorno de desarrollo, el cual se descarga del sitio web de Eclipse: http://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/kepler/SR1/eclipse-standard-kepler-SR1-win32.zip, dentro de este sitio escogemos el servidor de descarga.

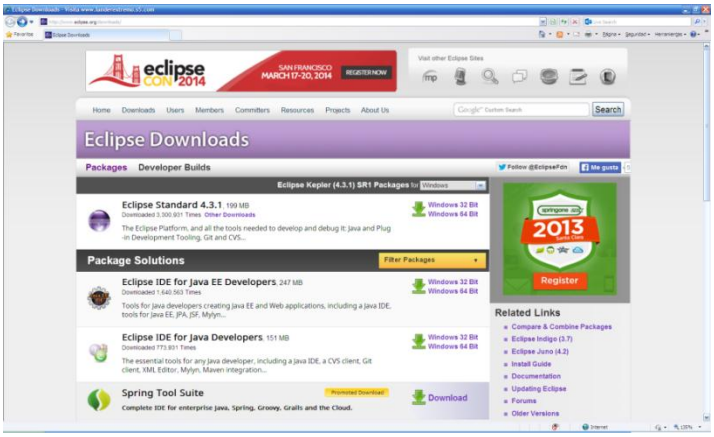


Figura 11. Sitio web de descarga del Eclipse Kepler 4.3.1

2. Una vez finalizada la descarga no se debe realizar ningún proceso de instalación, simplemente se debe descomprimir el fichero en un nuevo directorio y damos clic en el ejecutable de eclipse, la primera vez que se ejecuta la aplicación nos solicita escoger nuestro espacio de trabajo (*workspace),* carpeta donde se almacenarán todas las aplicaciones desarrolladas.

2.3.5. Instalar el plugin de Android para Eclipse

El siguiente paso es instalar el *plugin* específico de Android para la plataforma de desarrollo Eclipse Kepler, para esto nos valemos de la herramienta ADT *(Android Develoment Tools*) que nos facilita enormemente el desarrollo de proyectos basados en Android.

1. Seleccionamos en el Menú *Help>Install New Software*
2. Damos clic en agregar repositorio, le asignamos un nombre al repositorio que vamos a agregar (por ejemplo, Android) y en la localización ingresamos el siguiente repositorio web https://dl-ssl.google.com/android/eclipse/como se muestra en la Figura 12.

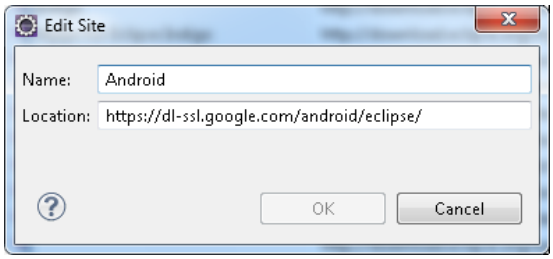


Figura 12. Agregación el repositorio de Android

3. Seleccionamos la pestaña *Develoment Tools* y procedemos a dar clic en siguiente, produciendo que se instalen paquetes necesarios para el desarrollo de software en Android. Una vez terminada la instalación se deberá reiniciar.

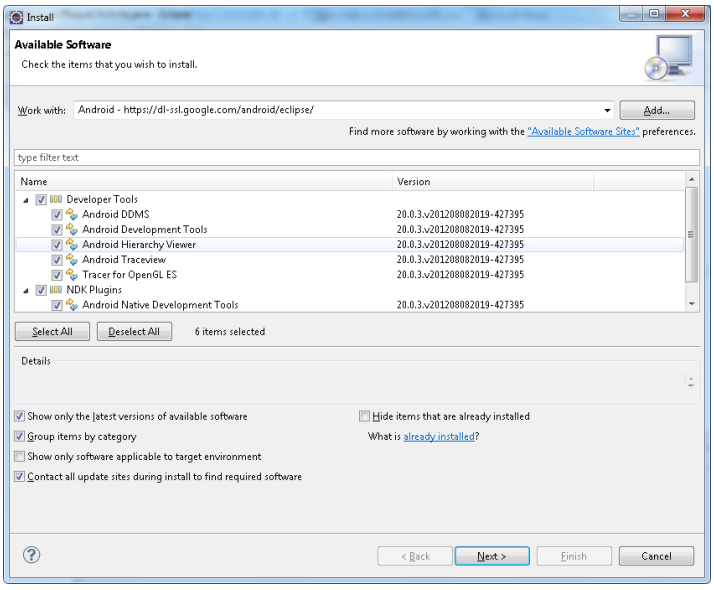


Figura 13. Selección e instalación de los paquetes de Android

Capítulo III: Metodologías de Desarrollo Ágil de Software

Se entiende como Desarrollo ágil de software a un paradigma de Desarrollo de Software basado en procesos ágiles. Los procesos ágiles de desarrollo de software, conocidos anteriormente como *metodologías livianas*, intentan evitar los tortuosos y burocráticos caminos de las metodologías tradicionales enfocándose en la gente y los resultados.

Es un marco de trabajo conceptual de la ingeniería de software que promueve iteraciones en el desarrollo a lo largo de todo el ciclo de vida del proyecto. Existen muchos métodos de desarrollo ágil; la mayoría minimiza riesgos desarrollando software en cortos lapsos de tiempo. El software desarrollado en una unidad de tiempo es llamado una iteración, la cual debe durar de una a cuatro semanas. Cada iteración del ciclo de vida incluye: planificación, análisis de requerimientos, diseño, codificación, revisión y documentación. Una iteración no debe agregar demasiada funcionalidad para justificar el lanzamiento del producto al mercado, pero la meta es tener un demo (sin errores) al final de cada iteración. Al final de cada iteración el equipo vuelve a evaluar las prioridades del proyecto.

Los métodos ágiles enfatizan las comunicaciones caras a cara en vez de la documentación. La mayoría de los equipos ágiles están localizados en una simple oficina abierta, a veces llamadas "plataformas de lanzamiento". La oficina debe incluir revisores, diseñadores de iteración, escritores de documentación y ayuda y directores de proyecto. Los métodos ágiles también enfatizan que el software funcional es la primera medida del progreso. Combinado con la preferencia por las comunicaciones cara a cara, generalmente los métodos ágiles son criticados y tratados como "indisciplinados" por la falta de documentación técnica.

3.1. Manifiesto Ágil

El Manifiesto Ágil comienza enumerando los principales valores del desarrollo ágil. Según el manifiesto se valora: [8]

- **Al individuo y sus interacciones** sobre los procesos y herramientas.
- **Al software que funciona** sobre la documentación extensiva.

- **La colaboración con el cliente** sobre la negociación contractual.
- **La respuesta ante el cambio** sobre el seguimiento de un plan.

Esto significa que aunque se valoran los elementos de la derecha, se valoran más los de la izquierda. Dichos valores sirven de inspiración a los doce principios enunciados por la Alianza Ágil:

1. Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.
2. Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.
3. Entregamos software funcional frecuentemente, en periodos entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.
4. Los responsables del negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.
5. Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo.
6. El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.
7. El software funcionando es la medida principal de progreso.
8. Los procesos ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.
9. La atención continua a la excelencia técnica y al buen diseño mejora la agilidad.
10. La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.
11. Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.
12. A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para ajustar y perfeccionar su comportamiento en consecuencia.

3.2. Scrum

Método de desarrollo ágil, que tiene sus orígenes alrededor de 1986 en un estudio sobre procesos de desarrollo utilizados en productos exitosos en Japón y los Estados Unidos, destacando su implementación en cámaras Canon, automóviles Honda, entre otros; pero no fue sino hasta 1993 que Jeff Sutherland, John Scumniotales y Jeff McKenna concibieron, ejecutaron y documentaron el primer Scrum para desarrollo ágil de software. Y finalmente en 1995 Ken Schwaber formalizó el proceso para la industria de desarrollo de software.

Scrum define métodos de gestión y control para complementar la aplicación de otros métodos ágiles como XP, en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Está pensada principalmente para proyectos con un rápido cambio de requisitos. [9]

3.2.1. Proceso

En Scrum, un proyecto se ejecuta en bloques temporales cortos y fijos (iteraciones de dos semanas, hasta un mes si así se necesita), los cuales son llamados sprints, donde cada iteración es un incremento ejecutable que proporciona un resultado completo, derivándose en un producto final que sea susceptible de ser entregado con el mínimo esfuerzo al cliente cuando lo solicite. La Figura 14 (Albaladejo, nd) representa gráficamente este proceso.

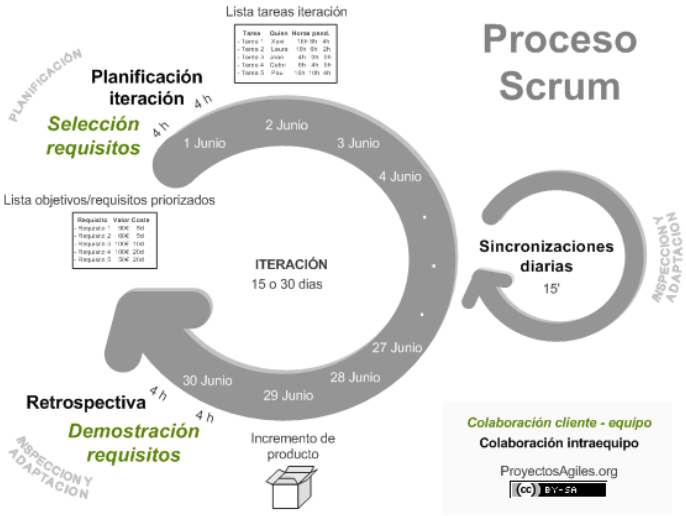


Figura 14. Proceso Scrum

3.2.2. Roles

En la Tabla 2 se muestran los roles que participan en el método Scrum.

Rol	Función
ScrumMaster	Actúa como facilitador. Mantiene los procesos y trabaja de forma similar al director del proyecto.
ProductOwner	Representa a los Stakeholders (clientes, interesados externos o internos).
Team	Incluye a los desarrolladores (de 5 a 9 personas).

Tabla 2 Roles de Scrum

3.2.3. Funcionamiento Básico

- El **ProductOwner** crea una lista de historias de usuario (requisitos de alto nivel priorizados) llamada **Product Backlog**.
- Durante la planeación **Sprint Planning**, el equipo selecciona un conjunto pequeño de requisitos del tope del product backlog, llamado **Sprint Backlog**, y decide cómo implementar esas tareas.
- El equipo tiene cierta cantidad de tiempo, llamado **Sprint**, para completar su trabajo.
- Usualmente el sprint es de 2 a 4 semanas, pero se realizan reuniones diarias de 15 minutos para llevar control del progreso y sincronizarse (**daily scrum**).
- Durante la ejecución del proyecto, el **ScrumMaster** mantiene al equipo enfocado en su objetivo.
- Al final de cada sprint, el trabajo debería ser potencialmente entregable, listo para entregárselo al cliente.
- El sprint culmina con un **sprint review** (demostración de los requisitos completados) y una retrospectiva (impresiones del sprint en general).
- Al inicio del siguiente sprint, el equipo selecciona otro subconjunto del product backlog y comienza a trabajar de nuevo. El ciclo se completa cuando suficientes ítems del backlog han sido completados o llegue la fecha final de entrega. Scrum asegura que el trabajo más importante ha sido completado cuando el proyecto termina.

3.3. Programación Extrema XP

La Programación Extrema proviene de sus siglas en inglés, eXtreme Programming (XP), y es un método de desarrollo ágil, formulado originalmente por Kent Beck, uno de los precursores del manifiesto ágil, en su libro titulado Extreme Programming Explained: Embrace Change (1999). XP se diferencia de las demás metodologías principalmente por el énfasisque hace sobre la adaptabilidad en vez de la previsibilidad. [10]

Según XP, con un poco de planificación, un poco de documentación, un poco de codificación y unas pocas pruebas instauradas antes y después de la codificación, se puede decidir si se está siguiendo un camino acertado o equivocado, evitando así tener que dar marcha atrás demasiado tarde.

3.3.1. Proceso de XP

Según refiere Beck (2000) el proceso de XP se presenta en estos cinco puntos.

1. El cliente define el valor de negocio a implementar.
2. El programador estima el esfuerzo necesario para su implementación.
3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
4. El programador construye ese valor de negocio.
5. Vuelve al paso 1.

3.3.2. Fases de XP

Beck (2000) presenta el ciclo de vida de XP mediante las siguientes fases citadas a continuación.

3.3.3. Fase de Exploración

Es la fase en la que se define el alcance general del proyecto. En esta fase, el cliente define lo que necesita mediante la redacción de sencillas “historias de usuario”. Los programadores estiman los tiempos de desarrollo en base a esta información. Debe quedar claro que las estimaciones realizadas en esta fase son primarias (ya que están

basadas en datos de muy alto nivel), y podrían variar cuando se analicen en más detalle en cada iteración. Esta fase dura típicamente un par de semanas, y el resultado es una visión general del sistema, y un plazo total estimado.

3.3.4. Fase de Planificación

La planificación es una fase corta, en la que el cliente, los gerentes y el grupo de desarrolladores acuerdan el orden en que deberán implementarse las historias de usuario, y, asociadas a éstas, las entregas. Típicamente esta fase consiste en una o varias reuniones grupales de planificación. El resultado de esta fase es un Plan de Entregas que se detallar en la sección Reglas y Prácticas.

3.3.5. Fase de Iteraciones

Esta es la fase principal en el ciclo de desarrollo de XP. Las funcionalidades son desarrolladas en esta fase, generando al final de cada una un entregable funcional que implementa las historias de usuario asignadas a la iteración. Como las historias de usuario no tienen suficiente detalle como para permitir su análisis y desarrollo, al principio de cada iteración se realizan las tareas necesarias de análisis, recabando con el cliente todos los datos que sean necesarios. El cliente, por lo tanto, también debe participar activamente durante esta fase del ciclo. Las iteraciones son también utilizadas para medir el progreso del proyecto. Una iteración terminada sin errores es una medida clara de avance.

3.3.6. Fase de Producción

Si bien al final de cada iteración se entregan módulos funcionales y sin errores, puede ser deseable por parte del cliente no poner el sistema en producción hasta tanto no se tenga la funcionalidad completa. En esta fase no se realizan más desarrollos funcionales, pero pueden ser necesarias tareas de ajuste.

3.3.7. Fase de Mantenimiento

Mientras la primera versión se encuentra en producción, el proyecto XP debe mantener el sistema en funcionamiento al mismo tiempo que desarrolla nuevas iteraciones. Para realizar esto se requiere de tareas de soporte para el cliente. De esta forma, la velocidad de desarrollo puede bajar después de la puesta del sistema en producción. La fase de mantenimiento puede requerir nuevo personal dentro del equipo y cambios en su estructura.

3.3.8. Fase Muerte del Proyecto

Cuando el cliente no tiene más historias para ser incluidas en el sistema. Esto requiere que se satisfagan las necesidades del cliente en otros aspectos como rendimiento y confiabilidad del sistema. Se genera la documentación final del sistema y no se realizan más cambios en la arquitectura. La muerte del proyecto también ocurre cuando el sistema no genera los beneficios esperados por el cliente o cuando no hay presupuesto para mantenerlo.

3.3.9. Reglas y Práctica

El aspecto más destacado de XP es la presentación de un conjunto de reglas y prácticas de una forma simple, siendo similar a las piezas de un rompecabezas, en la que existen muchas piezas pequeñas que lo conforman.(Wells, 2009). Vea Figura 15

Figura 15. Reglas y Practicas de Programación Extrema

3.3.10. Planeación

Hace posible una organización adecuada con el contacto continuo de todas las partes involucradas en el proyecto. Se inicia con la elaboración de las historias de usuario, las cuales sustituyen a los casos de uso, siendo dictaminadas por el cliente a manera de reflejar los requerimientos del sistema. De esta forma, se estima el tiempo de la planificación de entrega, y es a partir de esta última que se establece el proyecto en general, determinando un conjunto de reglas que permiten a cada integrante del proyecto tomar sus propias decisiones.

Todo esto con el fin de instaurar un plan o cronograma de entrega, para construir iteraciones que permitan realizar entregas frecuentes al cliente.

3.3.11. Diseño

Esta metodología hace énfasis en la simplicidad, ya que se afirma que un diseño simple toma mucho menos tiempo que uno complejo, y se busca la manera más simple de realizar los proyectos, haciéndolos más rápidos y menos costosos. XP hace uso de metáforas de sistema, de manera de hacer más viable y entendible el propósito del proyecto y a su vez tener suficiente contenido, guiando así la estructura y arquitectura del mismo. También se tiene la utilización de tarjetas de clases, responsabilidades y colaboración (CRC) para realizar el sistema como un equipo, permitiendo a la gente romper con la modalidad procedural del pensamiento y apreciar más la tecnología de objetos.

Se crean además soluciones spike, con el fin de averiguar respuestas a problemas técnicos o de diseño difíciles, de una manera simple y concreta. Se busca mantener el código listo para posibles cambios inesperados, existiendo una integración continua para reducir el impacto de agregar estas nuevas funcionalidades. Y por último, se debe refactorizar cuando y donde sea posible, a fin de eliminar redundancia, funcionalidades sin uso y lograr el rejuvenecimiento del diseño, ahorrando tiempo y aumentando la calidad.

3.3.12. Codificación

Gran parte del éxito de un proyecto XP se basa en que requiere indispensablemente la disponibilidad del cliente, quien conduce constantemente el trabajo, aportando un mayor valor agregado al equipo de desarrollo. Durante cada encuentro cara a cara se llega a un acuerdo con el cliente mediante una nueva selección de historias de usuarios que son incluidas en cada entrega programada. Además, se necesita de la presencia del cliente a la hora de realizar las pruebas funcionales.

Se deben seguir estándares y normas previamente acordados para la codificación, de manera de mantener un código consistente, legible y que pueda ser refactorizado. La realización de pruebas unitarias propuesta por XP implica un modelo inverso al que por lo general se conoce, ya que primero se crean las pruebas y después el código, ayudando a los desarrolladores a considerar realmente lo que hay que hacer, y estableciendo el desarrollo mínimo para pasar las pruebas antes definidas. Esto también proporciona una respuesta inmediata a medida que se está trabajando.

Uno de los principales factores que define a XP en su fase de codificación es que promueve la programación en pareja en una sola computadora, incrementando la calidad del software sin impactar el tiempo de entrega. Esto permite obtener un mejor diseño, mayor eficiencia en el código, menor cantidad de errores, mejor trabajo en equipo y mayor dinamismo.

La integración secuencial y continua en esta metodología promueve la publicación de las últimas versiones del sistema lo más pronto posible, siempre y cuando estén libres de errores. Esto permite mantener al sistema completamente integrado todo el tiempo, y así evitar retrasos innecesarios al trabajar con versiones antiguas del sistema.

XP remarca la propiedad colectiva del código, donde se anima a contribuir con nuevas ideas en cada uno de los segmentos del código, permitiendo poder cambiar cualquier parte de este en cualquier momento. El beneficio radica en el aumento de la calidad y reducción de errores.

3.3.13. Pruebas

En esta etapa de las reglas de XP, se hace especial énfasis a todo tipo de pruebas unitarias, siendo además la piedra angular de esta metodología. Primero se debe elegir un framework especializado en pruebas, luego se realiza la ejecución de todas las clases del sistema. Todos los módulos deben pasar las pruebas unitarias antes de ser liberados o publicados. De esta forma se debe resguardar el sistema y el conjunto de pruebas junto con el código de manera que otros desarrolladores puedan utilizarlo. En esta etapa se busca detectar cualquier tipo de error (bug), para corregirlo y depurarlo inmediatamente, tomando las medidas necesarias para que este no vuelva a suceder. Además de todo esto existen las llamadas pruebas de aceptación, que van estrechamente ligadas a las historias de usuario, en donde el cliente determina ciertos escenarios para poner a prueba al sistema, verificando la correctitud de respuesta del mismo.

3.3.14. Roles

La Tabla 3 presenta y describe los roles que toman parte en un desarrollo XP.

Rol	Función
Cliente	Escribe las historias de usuario y las pruebas funcionales para validar su implementación.
Programador	Genera las pruebas unitarias y desarrolla el código del sistema.
Encargado de Pruebas (Tester)	Ayuda al cliente a escribir las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.
Encargado de seguimiento (Tracker)	Verifica el grado de acierto entre las estimaciones realizadas y el tiempo real dedicado, para mejorar futuras estimaciones.
Entrenador (Coach)	Provee guías al equipo de forma que se apliquen prácticas XP y se siga el proceso correctamente.
Consultor	Miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto
Gestor (Big Boss)	Vínculo entre clientes y programadores, ayuda a que el equipo trabaje efectivamente creando las condiciones adecuadas. Su labor esencial es de coordinación.

Tabla 3 Roles de XP

Capítulo IV: Marco Aplicativo y Resultados

En el marco aplicativo de este trabajo especial de grado se presenta paso a paso cómo se llevaron a la práctica de conjunto de conceptos, procesos y tecnologías y así lograr los objetivos planteados al inicio de esta investigación.

Inicialmente se presenta la descripción del proceso de desarrollo a seguir y su adaptación a este trabajo de investigación. Y seguidamente las etapas del proceso, actividades, resultados, artefactos y estrategias involucradas en el desarrollo de la aplicación.

4.1. Adaptación de XP para este trabajo

En esta oportunidad, el proceso de desarrollo guía fue XP en combinación con Modelación ágil lo cual nos permite la realización de la parte práctica de nuestro trabajo de investigación de manera rápida, sencilla, eficiente y lo suficientemente documentada. A continuación se describe como se aplican las 4 fases de la Programación Extrema al desarrollo de nuestro sistema:

4.1.1. Planificación

El primer paso fue tomar unos días para realizar un análisis global del sistema que se quería construir, creando una representación sencilla de las partes que lo conformarían y cómo se comunicarían entre ellas. Esto sigue la idea de XP de crear una metáfora del sistema que represente de forma general cuál es el resultado que se persigue en el desarrollo.

Además, se creó una lista de historias de usuario durante la conversación inicial con el cliente, quien fue responsable de ordenarlas para su implementación siguiendo una prioridad determinada por el aporte de la historia de usuario a la funcionalidad principal del sistema: Luego la lista se dividió en dos grupos, uno con las historias de usuario y otro relacionado a la gestión de usuarios en el sistema. Se decidió trabajar en función del tiempo y así, cada historia de usuario tendría una cantidad de días estipulados (nunca mayor a tres (3) semanas o quince (15) días hábiles) para realizar actividades cortas que

en conjunto dieran como resultado la implementación de la funcionalidad descrita en la historia de usuario en cuestión.

Igualmente, se establecieron 3 valores para medir el riesgo de implementación de cada historia de usuario (bajo, medio y alto) determinado subjetivamente por los desarrolladores basándose en su experiencia con las tecnologías a utilizar y el trabajo que implicaba las actividades a realizar. El formato de presentación de las historias de usuario se observa en la tabla 4.

Número:	***Nombre:***	
Prioridad:	*Riesgo en desarrollo:*	*Estimación del tiempo de implementación:*
Descripción:		

Tabla 4 Formato para Historias de Usuario

Se realizó la planificación de iteraciones en las cuales se dividiría la implementación de las historias de usuario definidas, cada una con una duración de 3 a 4 semanas; y las entregas al cliente para realización de pruebas, correcciones y cambios se llevarían a cabo al término de cada iteración.

4.1.2. Diseño

Siguiendo las prácticas de Modelación Ágil, para cada iteración en la etapa de diseño se agregaron los diagramas o modelos que se creyeron necesarios para el entendimiento y la documentación del sistema. Y en caso de haberlos creado en alguna iteración anterior, se actualizaron, desechando las versiones anteriores.

4.1.3. Codificación

Se decidió un estándar sencillo para codificar tanto en la herramienta Visual Fox Pro, como en el lenguaje Java y php. Entre ellos se destaca: seguir la nomenclatura, declaración de variables, espaciado, y documentación.

4.1.4. Pruebas

Las pruebas unitarias se realizaron por historia de usuario, escribiendo los casos de prueba antes de comenzar a desarrollar.

Id Caso Prueba	Modulo	Descripción del Caso de Prueba	Resultado Esperado	Resultado Obtenido	Motivo de la Falla	Observaciones

Tabla 5 Formato de registro de pruebas unitarias del lado del cliente

4.2. Análisis Global del Sistema

Este análisis inicial parte de la propuesta realizada al inicio del documento y de la primera reunión con el cliente. De esta actividad se obtienen las historias de usuario y se construye un modelo del esquema de funcionamiento del sistema. Ambos resultados se presentan a continuación:

4.2.1. Historia de Usuarios

A continuación se presentan las Historias de Usuarios establecidas según formato de la tabla 4:

Número: 1	***Nombre: Interfaz de la Aplicación Móvil***	
Prioridad: Alta	*Riesgo en desarrollo: Bajo*	*Estimación del tiempo de implementación: 15 días*
Descripción: Crear una Aplicación para Android con las secciones necesarias para proveer al usuario la información y las herramientas para trabajar con ella.		

Tabla 6 Historia de Usuarios 1 Interfaz de la Aplicación Móvil

Número: 2	***Nombre: Módulo Registro del Dispositivo***	
Prioridad: Baja	*Riesgo en desarrollo: Baja*	*Estimación del tiempo de implementación: 2 días*
Descripción: El usuario debe proporcionar la información requerida para generar una clave manual o proporcionar el IMEI para su conformación vía Web.		

Tabla 7 Historia de Usuarios 2 Módulo Registro del Dispositivo

Número: 3	***Nombre: Módulo Configuración***	
Prioridad: Baja	*Riesgo en desarrollo: Baja*	*Estimación del tiempo de implementación: 2 día*
Descripción: El usuario debe proporcionar los parámetros de configuración concernientes a la identificación del vendedor, Dominio/IP, contador de pedidos, etc.		

Tabla 8 Historia de Usuarios 3 Módulo de Configuración

Número: 4	***Nombre: Módulo de Sincronización***	
Prioridad: Media	*Riesgo en desarrollo: Media*	*Estimación del tiempo de implementación: 2 días*
Descripción: El usuario podrá seleccionar sincronizar desde el portal de integración todos los Artículos, Clientes o Datos Recientes.		

Tabla 9 Historia de Usuarios 4 Módulo de Sincronización

Número:5	***Nombre: Módulo de Pedidos***	
Prioridad: Alta	*Riesgo en desarrollo: Medio*	*Estimación del tiempo de implementación: 5 días*
Descripción: El usuario debe seleccionar la empresa, el cliente, los ítems con su respectiva unidad y cantidad, una vez concluido se debe guardar y se podrá colocar una observación, y dará la opción de enviar o enviar después.		

Tabla 10 Historia de Usuarios 5 Módulo de Pedidos

Número: 6	***Nombre: Módulo Clientes***	
Prioridad: Media	*Riesgo en desarrollo: Media*	*Estimación del tiempo de implementación: 2 días*
Descripción: El usuario una vez seleccionada la empresa, debe poder visualizar una lista de clientes, los cuales al desplegarse debe mostrar la información del cliente, así como el saldo pendiente y sus cuentas por cobrar.		

Tabla 11 Historias de Usuarios 6 Módulo Clientes

Número: 7	***Nombre: Módulo Artículos***	
Prioridad: Media	*Riesgo en desarrollo: Medio*	*Estimación del tiempo de implementación: 2 días*
Descripción: El usuario una vez seleccionada la empresa, debe poder visualizar la lista de artículos disponible por la empresa, los cuales al desplegarse debe mostrar la información del artículo con su stock, precio, unidad de empaque y lote.		

Tabla 12 Historia de Usuarios 7 Módulo Histórico

Número: 8	***Nombre: Módulo Histórico***	
Prioridad: Media	*Riesgo en desarrollo: Alta*	*Estimación del tiempo de implementación: 5 días*
Descripción: El usuario debe seleccionar ver los pedidos por fecha o individual que son aquellos que corresponden al día en curso, para poder ver o retransmitir los mismos.		

Tabla 13 Historia de Usuarios 8 Módulo Histórico

Número: 9	***Nombre: Módulo Mantenimiento***	
Prioridad: Baja	*Riesgo en desarrollo: Baja*	*Estimación del tiempo de implementación: 1 día*
Descripción: El usuario podrá limpiar el temporal, históricos o todas las Base de datos locales pertenecientes a la aplicación móvil utilizada SQlite.		

Tabla 14 Historia de Usuarios 9 Módulo Mantenimiento

Número: 10	***Nombre: Aplicación Emisor***	
Prioridad: Alta	*Riesgo en desarrollo: Alta*	*Estimación del tiempo de implementación: 7 días*
Descripción: Esta aplicación será la encargada de actualizar la información de los clientes y artículos al portal de integración, se podrá definir los tiempos de envío, así como también se podrán asignar otros parámetros como almacenes, campo especial para filtrar artículos, campo del predio base, así como de los precios que se quieran subir, si se quieren filtrar solo los artículos con stock, y se desea excluir algún vendedor.		

Tabla 15 Historia de Usuarios 10 Aplicación Emisor

Número: 11	***Nombre: Aplicación Receptor***	
Prioridad: Alta	*Riesgo en desarrollo: Alta*	*Estimación del tiempo de implementación: 7 días*
Descripción: Esta aplicación será la encargada de recibir del portal de integración los pedidos y registrarlas al sistema Administrativo Profit 2K8 en forma de cotizaciones, para ello se podrá definir el intervalo de tiempo, así como el origen y destinos de los datos.		

Tabla 16 Historia de Usuarios 11 Aplicación Receptor

Número: 12	***Nombre: Portal de Integración***	
Prioridad: Alta	*Riesgo en desarrollo: Alta*	*Estimación del tiempo de implementación: 10 días*
Descripción: Este servidor será el encargado de intercambiar la información con la aplicación móvil, también contendrá la base de datos necesaria para interactuar tanto con la aplicación móvil, como con las aplicaciones emisor y receptor necesarias para la interacción con el sistema ERP Profit 2k8 Administrativo (con base de datos Sql Server)		

Tabla 17 Historia de Usuario 12 Portal de Integración

Del grupo anterior se seleccionaron como historias de usuario con mayor prioridad aquellas relacionadas a la construcción de interfaces de usuario (producto del sistema) y la interacción con el portal de integración. Estas se nombran a continuación:

HU 1.- Elaborar la Interfaz de la Aplicación Móvil.
HU 4.- Módulo de Sincronización.
HU 5.- Módulo de Pedidos.
HU 6.- Módulo Clientes.
HU 7.- Módulo Artículos.
HU 8.- Módulo Histórico.
HU 10.- Aplicación Emisor.
HU 11.- Aplicación Receptor.
HU 12.- Portal de Integración.

Las historias de usuario restantes fueron aquellas relacionadas con el manejo de los usuarios en el sistema y con la configuración:

HU 2.- Módulo Registro del Dispositivo.
HU 3.- Módulo Configuración.
HU 9.- Módulo Mantenimiento.

4.2.2. Arquitectura de la Solución

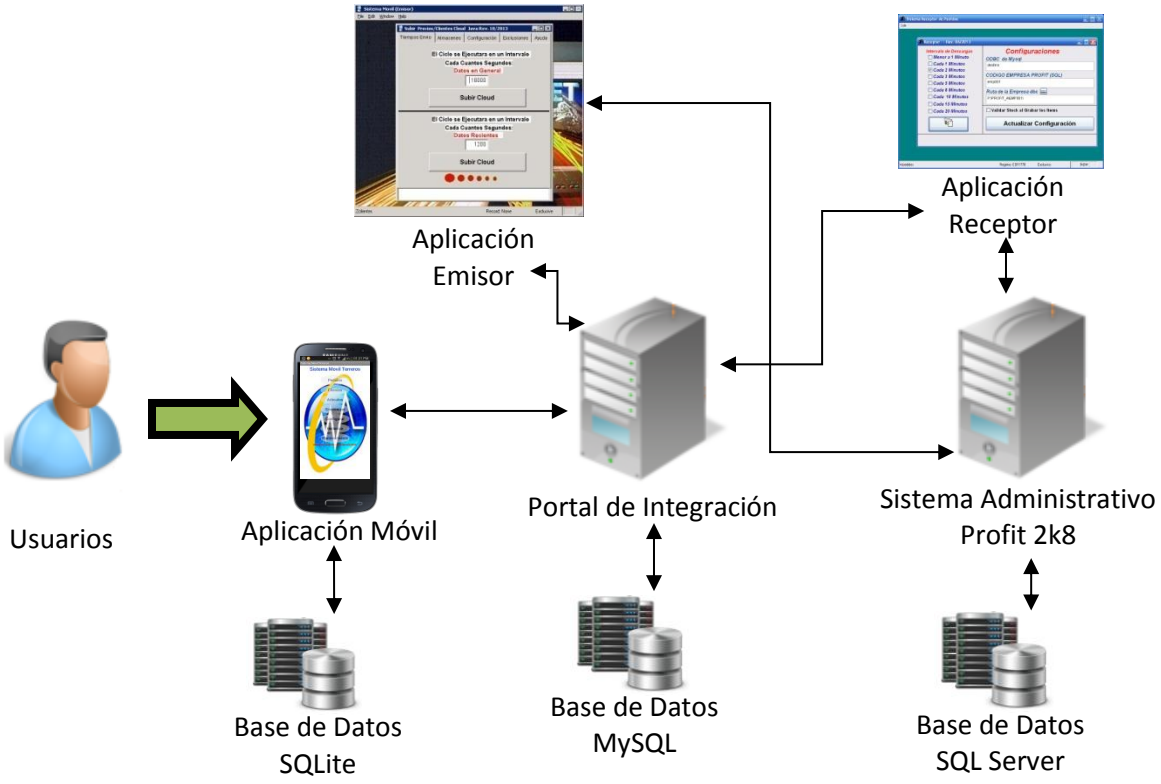


Figura 16. Arquitectura de la Solución

La presente arquitectura se desarrolló los componentes que constituyen la solución. Entre los componentes desarrollados se cuenta con: una interface denominada "Aplicación Emisor", programado con la herramienta Visual Fox Pro, cuya función principal es recopilar la información contenida en el Sistema Administrativo Profit 2K8, almacenada en una base de datos SQL Server, la cual es posteriormente transferida al componente "Portal de Integración" que almacena la información en una base de datos MySql. El tercer componente de la arquitectura desarrollada es la "Aplicación Móvil"; adicionalmente permite consultar los artículos con su stock aproximados, ya que la aplicación móvil cuenta con la capacidad de sincronizar los artículos cargados previamente con el Portal de Integración, el cual es actualizado por la Aplicación Emisor.

Y por último se encuentra el componente "Aplicación Receptor" el cual es el encargado de descargar los pedidos registrados por la fuerza de venta en el Portal de Integración a través de la aplicación Móvil, para luego registrarlos en el Sistema Administrativo Profit 2K8.

4.2.3. Especificaciones Técnicas

El desarrollo del sistema se implantará utilizando las siguientes herramientas:

- Servidor Web: Apache 2.2.22 (Win64) y PHP 5.3.13
- Entorno de desarrollo JDK 8 en Windows.
- Eclipse Kepler 4.3.1.
- Visual Fox Pro Versión 9
- Manejador de Base de Datos Mysql Server ver. 5.5.24
- Manejador de Base de Datos Microsoft SQL Server 2008 ver. 10.0.1600

Una vez establecidos los requerimientos iníciales, seleccionadas las primeras historias de usuario a implementar y haber modelado una idea general de cómo funcionaría el sistema, se planifican 4 iteraciones.

4.3. Primera Iteración

4.3.1. Planificación

Iteración 1	
Descripción	Desarrollo de la aplicación que el usuario utilizará para realizar sus pedidos, incluyendo los procesos de consulta de clientes, artículos y pedidos, así como sincronización, configuración y mantenimiento.
Historias de Usuario a Desarrollar	1.- Elaborar interfaz de la aplicación Móvil.
	5.- Módulo de Pedidos
	6.- Módulo de Clientes
	7.- Módulo de Artículos
	8.- Módulo de Histórico
	2.- Registro del Dispositivo
	4.- Módulo de Sincronización
	3.- Módulo de Configuración
	9.- Módulo de Mantenimiento
Tiempo Estimado	24 días

Tabla 18 Iteración 1

4.3.2. Tareas por Historia de Usuario

HU 1.- Elaborar la Interfaz de la Aplicación Móvil.

- Hacer un diseño base de la aplicación.
- Agregar el contenido básico: título, logo, secciones y versión.

HU 5.- Módulo de Pedidos.

- Elaborar una lista con las empresas definidas.
- Ubicar los controles para seleccionar el cliente y los artículos por pedido.
- Permitir ver el detalle del pedido y dar la posibilidad de eliminar ítems creados.
- Dar la posibilidad al usuario de al guardar el pedido poder agregar algún comentario y enviar el pedido.

HU 6.- Módulo de Clientes.

- Acceder a la empresa mediante una lista de empresas definidas.
- Generar un listado con los clientes filtrados por vendedor.
- Mostrar por cada cliente su código, descripción, dirección, teléfonos, RIF, región, saldo y cuentas por cobrar.
- Permitir realizar una búsqueda por descripción o nombre de la empresa.

HU 7.- Módulo de Artículos.

- Acceder a la empresa mediante una lista de empresas definidas.
- Generar un listado con los artículos.
- Mostrar por cada artículo su código, descripción, stock actual, precio actual, unidad de empaque y lote.
- Permitir realizar una búsqueda por descripción o nombre del artículo.

HU 8.- Módulo de Histórico.

- Solicitar al usuario que tipo de filtro desea hacer por fecha o mostrar todos los pedidos realizados.
- Mostrar un listado con los pedidos indicando el número de pedido, código del cliente, descripción, fecha y los artículos.
- Al seleccionar él o los pedidos, permitir retransmitirlo.

HU 2.- Registro del Dispositivo.

- Solicitar al usuario una clave de registro.
- Permitir el registro de forma manual o mediante un registro vía Web.

HU 4.- Módulo de Sincronización.

- Realizar los botones para sincronizar Artículos, Clientes o Artículos y Clientes del día.
- Almacenar en la base de datos local la información proveniente del portal de integración.

HU 3.- Módulo de Configuración.

- Solicitar al usuario informaciones relevantes al código y nombre del vendedor.
- Establecer la ruta de almacenamiento en el portal de integración.
- Especificar el contador de pedido para evitar la duplicidad.
- Establecer checklist con los parámetros adicionales de validación.

HU 9.- Módulo de Mantenimiento.

- Encargado de proveer al usuario opciones para manipular los datos almacenados temporalmente en el teléfono.
- Establecer botones para vaciar datos temporales, históricos o todas las bases de datos.

4.3.3. Diseño

A partir de los componentes a implementar para el sistema, se investigó cuáles de los paquetes básicos de Android se utilizarán para crear la actividad y trabajar con vistas que a su vez contendrán los elementos necesarios para la interacción con el usuario. Los colores y la apariencia su realizada bajo la sugerencia de la directiva de la empresa.

Figura 17. Diseño de la Aplicación Móvil

4.3.4. Codificación

La Interfaz inicial de la aplicación es la encargada de validar los permisos e iniciar la conexión con la base de datos interna, Android requiere un archivo de manifiesto AndroidManifest que describe el contexto de la aplicación y sus actividades, proveedores de contenido, así como los permisos, en ella contiene la actividad inicial llamado ConeccionsqliteActivity.Java. A continuación se mostrarán algunas de las funciones o procedimientos más relevantes a cada Actividad.

Validación de Licencia	Codificación
 **La Clase ConecctionsqliteActivity** Extiende de la clase Activity en esta clase se definen las variables para poder verificar la licencia de la aplicación. El método setContentView en el procedimiento onCreate se crea la interfaz del usuario que es un diseño definido en main.xml en el directorio /res/layout.	`//Se omiten varias importaciones.` `public class ConeccionsqliteActivity extends Activity` `{ public static int sw = 0;` `String imei;` `BaseDatosHelper miBBDDHelper;` `public int xfiltro = 2;` `public String xstatus = "";` `public void onCreate(Bundle paramBundle)` `{actualizabd();` `licencia();` `if (sw == 1)` `{ super.onCreate(paramBundle);` `if (tablet == 0)` **`setContentView`**`(2130903055);}` `….. (abreviado)` `public void licencia()` `{crearBBDD();` `llamarven();` `String[] arrayOfString = { " " };` `Cursor localCursor = this.miBBDDHelper.myDataBase.` `rawQuery("select serial from vendedor` `where co_ven <>?", arrayOfString);` `String str1 = String.valueOf(Long.valueOf(()(1.017D` `*Long.valueOf(Long.valueOf(((TelephonyManager)` `getSystemService("phone")).getDeviceId()).` `longValue() / 2L).longValue())));` `if (localCursor.moveToFirst())` `{String str2 = localCursor.getString(0);` `double d = Double.valueOf(str1).doubleValue();` `if (Double.valueOf(str2).doubleValue() == d)` `{Log.i("serial ", "correcto");` `sw = 1;` `this.miBBDDHelper.myDataBase.close();}` `}` `else` `{return;}` `this.miBBDDHelper.myDataBase.close();` `Log.i("serial ", "Incorrecto ");` `Bundle localBundle = new Bundle();` `Intent localIntent = new Intent(getBaseContext(),` `serial.class);` `localIntent.putExtras(localBundle);` `startActivityForResult(localIntent, 0);` `}`

Tabla 19 Codificación de ConecctionsqliteActivity.java

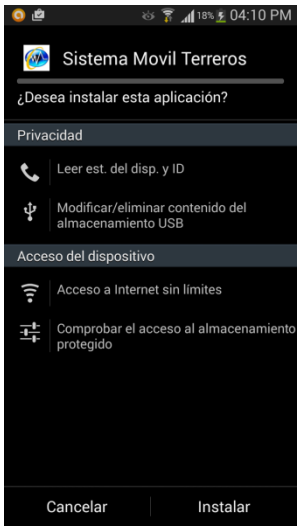
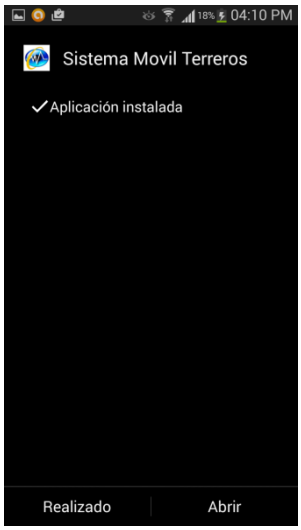
Sistema Móvil	Codificación
 Contiene los descriptores de las clases de nivel superior de la aplicación. Aquí se combinan todos los detalles de una aplicación, la información necesaria para ejecutar la aplicación, así como los Intent que puede procesar y los permisos que necesita 	**AndroidManifest.xml** `<?xml version="1.0" encoding="utf-8" standalone="no"?>` `<manifest` `xmlns:android="http://schemas.android.com/apk` `/res/android" package="fuentes.conecion">` `<uses-permission android:name="android.permission.` `WRITE_EXTERNAL_STORAGE"/>` `<uses-permission android:name="android.permission.` `INTERNET"/>` `<uses-permission android:name="android.permission.` `READ_PHONE_STATE"/>` `<supports-screens android:anyDensity="true"` `android:largeScreens="true"` `android:normalScreens="true"` `android:resizeable="true"` `android:smallScreens="true"/>` `<application android:debuggable="true"` `android:icon="@drawable/ic_launcher"` `android:label="@string/app_name">` `<activity android:label="@string/app_name"` `android:name=".ConeccionsqliteActivity">` `<intent-filter>` `<action android:name="android.intent.` `action.MAIN"/>` `<category android:name="android.intent.` `category.LAUNCHER"/>` `</intent-filter>` `</activity>` `<activity android:name=".articulosexpandible"/>` `<activity android:name=".BaseDatosHelper"/>` `<activity android:name=".camposclientes"/>` `<activity android:name=".clientesexpandible"/>` `<activity android:name=".ControladorLista"/>` `<activity android:name=".Sincronizar"/>` `<activity android:name=".serial"/>` `<activity android:name=".pedidos"/>` `<activity android:name=".mantenimiento"/>` `<activity android:name=".listaclientes"/>` `<activity android:name=".listaart"/>` `<activity android:name=".historico"/>` `<activity android:name=".filtroempresa"/>` `<activity android:name=".filtrofecha"/>` `<activity android:name=".historicoxfecha"/>` `<activity android:name=".FiltroempreSaclientes` `expandible"/>` `<activity android:name=".Filtroempresaarticulos` `expandible"/>` `</application>` `</manifest>`

Tabla 20 Codificación AnfroidManifest.xml

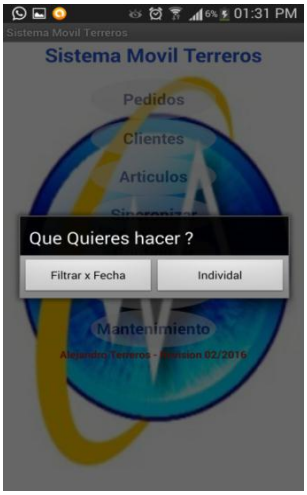
Seleccionar Empresas	Codificación
 Contiene elementos de la interfaz de usuario. De allí también se desprende la Validación del Histórico mediante un proceso preguntar() que dada la pregunta de desea filtrar por Fecha o Individual, el sistema invoca a las clases filtrofecha.class o histórico.class. 	**Main.xml** `<?xml version="1.0" encoding="utf-8"?>` `<LinearLayout android:gravity="fill"` `android:orientation="vertical"` `android:background="@drawable/fondo5"` `android:layout_width="fill_parent"` `android:layout_height="fill_parent"` `xmlns:android="http://schemas.android.com/apk/res/` `android">` `<TextView` `Android:textAppearance="?android:textAppearanceLarge"` `android:textSize="25.0dip" android:textStyle="bold"` `android:textColor="@drawable/blue"` `android:layout_gravity="center"` `android:id="@id/textView1"` `android:layout_width="wrap_content"` `android:layout_height="wrap_content"` `android:text="Sistema Movil Terreros" />` `<TextView` `android:textAppearance="?android:textAppearanceLarge"` `android:id="@id/textView6"` `android:layout_width="wrap_content"` `android:layout_height="15.0dip" android:text="  " />` `<Button android:textSize="20.0dip"` `android:textStyle="bold"` `android:textColor="#ff080e4e"` `android:layout_gravity="center_horizontal"` `android:id="@id/button1"` `android:background="@drawable/boton_circular"` `android:layout_width="150.0dip"` `android:layout_height="53.0dip"` `android:text="Pedidos" android:alpha="0.5" />` `<Button android:textSize="20.0dip"` `android:textStyle="bold"` `android:textColor="#ff080e4e"` `android:layout_gravity="center_horizontal"` `android:id="@id/button2"` `android:background="@drawable/boton_circular"` `android:layout_width="150.0dip"` `android:layout_height="50.0dip"` `android:text="Clientes" android:alpha="0.5" />` …… `<TextView` `android:textAppearance="?android:textAppearanceMedium"` `android:textSize="13.0dip" android:textStyle="bold"` `android:textColor="@drawable/red"` `android:layout_gravity="center"` `android:id="@id/textView2"` `android:layout_width="wrap_content"` `android:layout_height="wrap_content"` `android:text="Alejandro Terreros - Revision` `02/2016"/>` `</LinearLayout>`

Tabla 21 Pantalla Inicial Main.xml

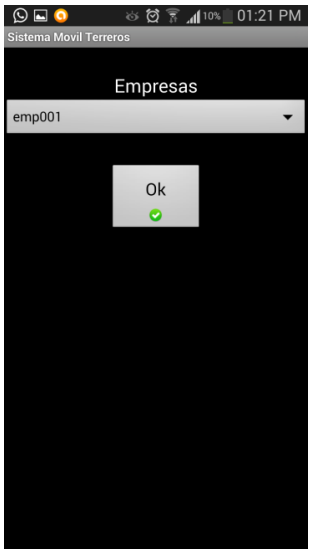
Seleccionar Empresas	Codificación
 Una vez seleccionado el botón Artículo, Cliente o Pedido se solicita al usuario que seleccione la empresa a utilizar dado que el sistema Profit 2k8 es multi-empresa puede un vendedor de la fuerza de ventas realizar los pedidos a cualquiera de las empresas disponibles.	`public void onCreate(Bundle paramBundle)` `  {` `    super.onCreate(paramBundle);` `    setContentView(2130903045);` `    llamarven();` `    String[] arrayOfString = { " " };` `    final Cursor localCursor =` `this.miBBDDHelper.myDataBase.rawQuery("select empresa as` `_id,empresa from articulos where empresa <>? group by` `empresa", arrayOfString);` `if (localCursor.moveToFirst())` `    {` `      xmensaje = "Si Existen Empresas ";` `Spinner localSpinner = (Spinner)findViewById(2131034141);` `      SimpleCursorAdapter localSimpleCursorAdapter = new` `SimpleCursorAdapter(this, 17367048, localCursor, new` `String[] { "empresa" }, new int[] { 16908308 });` `localSimpleCursorAdapter.setDropDownViewResource(17367049)` `;` `      localSpinner.setAdapter(localSimpleCursorAdapter);` `localSpinner.setOnItemSelectedListener(new` `AdapterView.OnItemSelectedListener()` `      {` `        public void onItemSelected(AdapterView<?>` `paramAnonymousAdapterView, View paramAnonymousView, int` `paramAnonymousInt, long paramAnonymousLong)` `{` `          int i = localCursor.getPosition();` `          localCursor.moveToFirst();` `          localCursor.moveToPosition(i);` `          filtroempresa.xempresa =` `localCursor.getString(1);` `}` `        public void onNothingSelected(AdapterView<?>` `paramAnonymousAdapterView)` `        {` `        }` `      });` `    }` `((Button)findViewById(2131034143)).setOnClickListener(new` `View.OnClickListener()` `    {` `      public void onClick(View paramAnonymousView)` `      {` `        localCursor.close();` `filtroempresa.this.miBBDDHelper.myDataBase.close();` `        Bundle localBundle = new Bundle();` `        Intent localIntent = new` `Intent(paramAnonymousView.getContext(), pedidos.class);` `localIntent.putExtras(localBundle);` `filtroempresa.this.startActivityForResult(localIntent, 0);` `      }` `    });` `  }`

Tabla 22 Filtroempresa.java

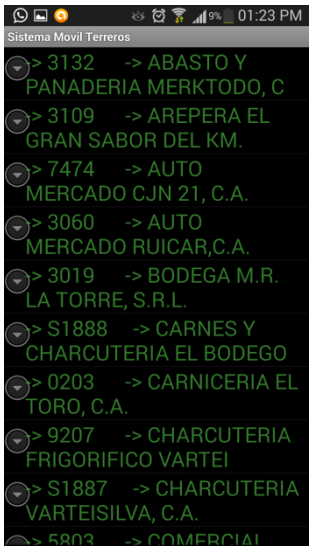
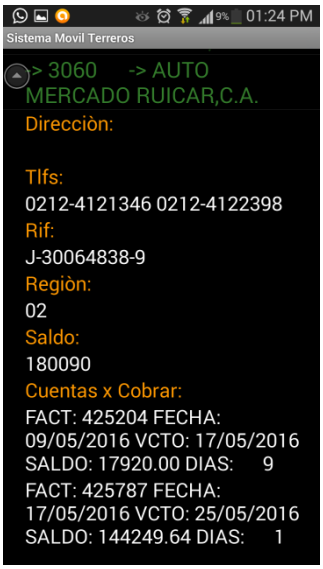
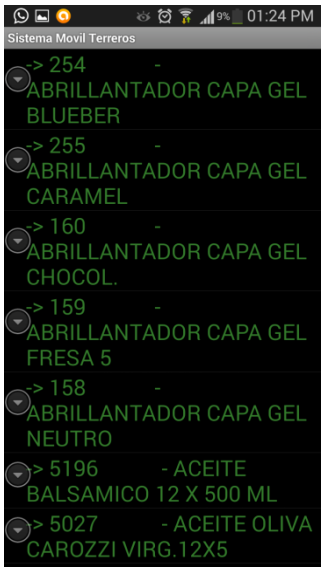
Clientes	Codificación
 En el Activity Clientes, el procedimiento pasardata() selecciona de la base de datos local los Clientes disponibles previamente sincronizados y los muestra en una lista desplegable en un objeto spinner en Android. 	``` public void pasardata() { String[] arrayOfString = new String[1]; arrayOfString[0] = ("%" + xempresa + "%"); Cursor localCursor1 = this.miBBDDHelper.myDataBase.rawQuery("select co_cli as _id,cli_des,co_cli,co_ven,direcc,rif,saldo,tlf from clientes where empresa like? order by cli_des", arrayOfString); this.miBBDDHelper.myDataBase.delete("clientesT", null, null); int i = 0; ContentValues localContentValues1 = new ContentValues(); localCursor1.moveToFirst(); Cursor localCursor2; int j; ContentValues localContentValues2; if (localCursor1.isAfterLast()) { localCursor2 = this.miBBDDHelper.myDataBase.rawQuery("select tasa_iva,tipo_imp,art_des,co_art,co_lin,prec_vta, stock,unidad,pvp1,pvp2,pvp3,pvp4,empresa,xdetalle, modelo from articulos where empresa like? order by art_des ", arrayOfString); …… } localContentValues2.put("tasa_iva", str7); localContentValues2.put("tipo_imp", str8); localContentValues2.put("art_des", str9); localContentValues2.put("co_art", str10); localContentValues2.put("co_lin", str11); localContentValues2.put("prec_vta", Double.valueOf(d2)); localContentValues2.put("stock", Double.valueOf(d3)); localContentValues2.put("unidad", str12); localContentValues2.put("pvp1", Double.valueOf(d4)); localContentValues2.put("pvp2", Double.valueOf(d5)); localContentValues2.put("pvp3", Double.valueOf(d6)); localContentValues2.put("pvp4", Double.valueOf(d7)); localContentValues2.put("empresa", str13); localContentValues2.put("xdetalle", str14); localContentValues2.put("modelo", str15); this.miBBDDHelper.getWritableDatabase(); this.miBBDDHelper.myDataBase.insert("articulosT", null, localContentValues2); localCursor2.moveToNext(); } } } ```

Tabla 23 Codificación Clientes.java

Artículos	Codificación



En el Activity Artículo el procedimiento pasardata() se selecciona de la base de datos local los Artículos disponibles previamente sincronizados y los muestra en una lista desplegable en un objeto spinner en Android.


```
public void pasardata()
{ String[] arrayOfString = new String[1];
    arrayOfString[0] = ("%" + xempresa + "%");
    Cursor localCursor1 = this.miBBDDHelper.myDataBase.rawQuery("select co_cli as
_id,cli_des,co_cli,co_ven,direcc,rif,saldo,tlf from clientes where empresa like? order
by cli_des", arrayOfString);
    this.miBBDDHelper.myDataBase.delete("clientesT", null, null);
    int i = 0;
    ContentValues localContentValues1 = new ContentValues();
localCursor1.moveToFirst();
    Cursor localCursor2;
    int j;
ContentValues localContentValues2;
    if (localCursor1.isAfterLast())
    {
      localCursor2 = this.miBBDDHelper.myDataBase.rawQuery("select
tasa_iva,tipo_imp,art_des,co_art,co_lin,prec_vta,stock,unidad,pvp1,pvp2,pvp3,pvp4,empr
esa,xdetalle,modelo from articulos where empresa like? order by art_des ",
arrayOfString);
      this.miBBDDHelper.myDataBase.delete("articulosT", null, null);
      j = 0;
      localContentValues2 = new ContentValues();
      localCursor2.moveToFirst();
    }
    while (true)
    {
      if (localCursor2.isAfterLast())
      {
        this.miBBDDHelper.myDataBase.close();
        localCursor2.close();
        localCursor1.close();
        return;
        String str1 = localCursor1.getString(1);
        String str2 = localCursor1.getString(2);
        String str3 = localCursor1.getString(3);
        String str4 = localCursor1.getString(4);
        String str5 = localCursor1.getString(5);
        double d1 = localCursor1.getDouble(6);
        String str6 = localCursor1.getString(7);
        i++;
        localContentValues1.put("cli_des", str1);
        localContentValues1.put("co_cli", str2);
        localContentValues1.put("co_ven", str3);
        localContentValues1.put("direcc", str4);
        localContentValues1.put("rif", str5);
        localContentValues1.put("saldo", Double.valueOf(d1));
        localContentValues1.put("tlf", str6);
        this.miBBDDHelper.getWritableDatabase();
        this.miBBDDHelper.myDataBase.insert("clientesT", null, localContentValues1);
        localCursor1.moveToNext();
        break;
      }
      String str7 = localCursor2.getString(0);
      String str8 = localCursor2.getString(1);
      String str9 = localCursor2.getString(2);
      String str10 = localCursor2.getString(3);
      String str11 = localCursor2.getString(4);
      double d2 = localCursor2.getDouble(5);
      double d3 = localCursor2.getDouble(6);
      String str12 = localCursor2.getString(7);
      double d4 = localCursor2.getDouble(8);
      double d5 = localCursor2.getDouble(9);
      double d6 = localCursor2.getDouble(10);
      double d7 = localCursor2.getDouble(11);
      String str13 = localCursor2.getString(12);
      String str14 = localCursor2.getString(13);
      String str15 = localCursor2.getString(14);
      j++;
      localContentValues2.put("tasa_iva", str7);
      localContentValues2.put("tipo_imp", str8);
      localContentValues2.put("art_des", str9);
      localContentValues2.put("co_art", str10);
      localContentValues2.put("co_lin", str11);
      localContentValues2.put("prec_vta", Double.valueOf(d2));
      localContentValues2.put("stock", Double.valueOf(d3));
      localContentValues2.put("unidad", str12);
      localContentValues2.put("pvp1", Double.valueOf(d4));
      localContentValues2.put("pvp2", Double.valueOf(d5));
      localContentValues2.put("pvp3", Double.valueOf(d6));
      localContentValues2.put("pvp4", Double.valueOf(d7));
      localContentValues2.put("empresa", str13);
      localContentValues2.put("xdetalle", str14);
      localContentValues2.put("modelo", str15);
      this.miBBDDHelper.getWritableDatabase();
      this.miBBDDHelper.myDataBase.insert("articulosT", null, localContentValues2);
localCursor2.moveToNext();
    }
  }
}
```

Tabla 24 Código de Articulos.java

Sincronizar	Codificación
 El Activity Sincronizar Inicia un procedimiento de descarga para obtener del portal de integración los Artículos o Clientes según se seleccione o los nuevos Artículos y Clientes del día.	`public void onCreate(Bundle paramBundle)` `  {` `    super.onCreate(paramBundle);` `    setContentView(2130903063);` `    final Button localButton1 =` `(Button)findViewById(2131034155);` `    localButton1.setOnClickListener` `(new View.OnClickListener()` `    {` `public void onClick(View` `paramAnonymousView)` `{` `localButton1.setBackgroundColor(-16711681);` `localButton1.setEnabled(false);` `   Sincronizar.this.abrirbd();` `   Sincronizar.this.xurl();` `   new Thread(new Runnable()` `        {` `          public void run()` `          {` `           Sincronizar.this.abrirbd();` `           Sincronizar.this.sincronizarart();` `           Sincronizar.this.abrirbd();` `Sincronizar.this.sincronizarart1();` `           Sincronizar.this.abrirbd();` `           Sincronizar.this.sincronizarart2();` `           Sincronizar.this.abrirbd();` `           Sincronizar.this.sincronizarart3();` `           this.val$spinnerDialog.dismiss();` `           Log.i("FINNNNN 88888", "");` `           System.exit(0);` `          }` `        }).start();` `      }` `    });`

Tabla 25 Código de Sincronizar.java

Configurar	Codificación
 El Activity Configurar Toma del archivo de Base de Datos SQLite llamado “movil”, los campos necesarios para utilizarlos como parámetros de usuario y de configuración inicial de la Aplicación Móvil. Adicionalmente, permite al cliente actualizarla parasu correcto funcionamiento.	public class BaseDatosHelper extends SQLiteOpenHelper { private static String DB_NAME = "movil"; private static String DB_PATH = "/data/data/fuentes.conecion/databases/"; static String xmensaje; private final Context myContext; public SQLiteDatabase myDataBase; public BaseDatosHelper(Context paramContext) { super(paramContext, DB_NAME, null, 1); this.myContext = paramContext;} private boolean comprobarBaseDatos() { try {SQLiteDatabase localSQLiteDatabase2 = SQLiteDatabase.openDatabase(DB_PATH + DB_NAME, null, 1); localSQLiteDatabase1 = localSQLiteDatabase2; if (localSQLiteDatabase1 != null) localSQLiteDatabase1.close(); return localSQLiteDatabase1 != null; } catch (SQLiteException localSQLiteException) { while (true) SQLiteDatabase localSQLiteDatabase1 = null;} } private boolean comprobarBaseDatos1() { try {SQLiteDatabase localSQLiteDatabase2 = SQLiteDatabase.openDatabase(DB_PATH + DB_NAME, null, 0); localSQLiteDatabase1 = localSQLiteDatabase2; boolean bool = false; if (localSQLiteDatabase1 != null) bool = true; return bool; } catch (SQLiteException localSQLiteException) { while (true) SQLiteDatabase localSQLiteDatabase1 = null; } }

Tabla 26 Código de BaseDatosHelper.java

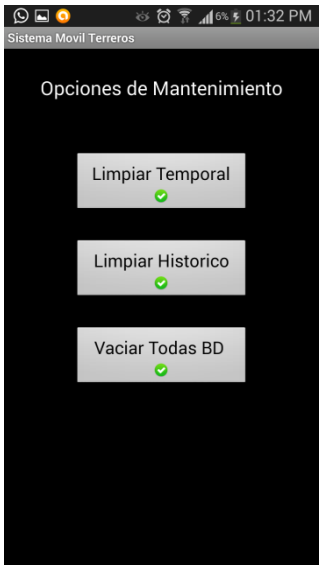
Mantenimiento	Codificación
 En el siguiente Activity Mantenimiento; contiene todos los procedimientos necesarios para limpiar la data que se encuentra almacenada localmente en la base de datos Móvil (SQLite).	`public class mantenimiento extends Activity` `{` `  static String xmensaje;` `  BaseDatosHelper miBBDDHelper;` `public void abrirbd()` `  {` `    this.miBBDDHelper = new BaseDatosHelper(this);` `    try` `    {` `      this.miBBDDHelper.crearDataBase();` `      this.miBBDDHelper.abrirBaseDatosV();` `      return;` `    }` `    catch (IOException localIOException)` `    {` `      while (true)` `        localIOException.printStackTrace();` `    }` `  }` `public void actualizarbd()` `  {` `    this.miBBDDHelper = new BaseDatosHelper(this);` `    this.miBBDDHelper.onUpgrade1();` `  }` `public void borrar()` `  {` `    this.miBBDDHelper.myDataBase.delete("temporal",null, null);` `    this.miBBDDHelper.myDataBase.close();` `  }` `public void borrar1()` `  {` `    this.miBBDDHelper.myDataBase.delete("pedidos", null,` `null);` `    this.miBBDDHelper.myDataBase.close();` `  }` `public void borrar3()` `  {` `    this.miBBDDHelper.myDataBase.delete("lin_art", null,null);` `    this.miBBDDHelper.myDataBase.delete("articulos",null,null);` `    this.miBBDDHelper.myDataBase.delete("tmppvp", null,null);` `    this.miBBDDHelper.myDataBase.delete("almacenes",null,null);` `    this.miBBDDHelper.myDataBase.delete("temporal",null,null);` `    this.miBBDDHelper.myDataBase.delete("pruebas", null, null);` `    this.miBBDDHelper.myDataBase.delete("pedidos", null,null);` `    this.miBBDDHelper.myDataBase.delete("clientes",null, null);` `    this.miBBDDHelper.myDataBase.delete("lotes", null, null);` `this.miBBDDHelper.myDataBase.close();` `  }`

Tabla 27 Código de Mantenimiento.java

4.3.5. Pruebas

Se realizaron un conjunto de pruebas, para verificar la operatividad de los Actividades, como interactuar entre los integración entre la Aplicación Móvil y el Portal de Integración.

Id Caso Prueba	Modulo	Descripción del Caso de Prueba	Resultado Esperado	Resultado Obtenido	Motivo de la Falla	Observaciones
1	***Interfaz de la Aplicación Móvil***	Colocar todos los botones con las actividades que se pueden realizar	Dar una apariencia visual practica al usuario	Se colocaron todos los botones de comando		Se generó un botón circular a través de un Shape tipo Oval
2	***Interfaz de la Aplicación Móvil***	Colocar un mecanismo de registro del dispositivo	Registrar el dispositivo manual o por IMEI	Se registra el dispositivo mediante una clave o por el portal		Se debe registrar el IMEI en portal de integración o generar una clave de conformación
3	***Módulo de Artículos***	Generar un lista con los Artículos mostrando información relevante al usuario	Listar la Información mediante una lista de listas desplegables	El Listado de los artículos con sus características más importantes	Carencia de filtrado	Se agregó por sugerencia del cliente un filtro por descripción del artículo
4	***Módulo de Pedidos***	Generar un pedido	Seleccionar el cliente y cada uno de los artículos	El usuario genera un pedido el cual al guardarse solicita al vendedor si este desea enviarlo	Carencia de Observaciones	Se adicionó una observación por parte del vendedor al momento del envío del pedido

Tabla 28 Pruebas

4.3.6. Observaciones

Durante las pruebas de aceptación con el cliente, éste realizó correcciones acerca de algunos parámetros de su apariencia tales como: color de fondo de la Aplicación, color de la fuente y definición del área de trabajo.

Gran parte de la plataforma Android gira en torno a la interfaz de usuario y a los conceptos de actividades y vistas, así mismo la Aplicación realiza peticiones a otras aplicaciones mediante invocaciones Intent:Activity.

Una de las ventajas de la plataforma Android es la incorporación de una base de datos relacional. SQLite ofrece todo lo necesario para el almacenamiento local de datos, a la vez que resulta rápido y sencillo.

Se definió una clase BaseDatosHelper con las constantes que definen los valores estáticos relacionados con la base de datos que estamos trabajando “movil”. Definiendo métodos para la creación, actualización y apertura de la base de dato. Es importante mencionar la utilización del objeto bundle utilizado para pasar datos entre los activity.

Por último, para el proceso de la sincronización se utilizó la API HttpClient de Apache, y se generó la solicitud de red en un Thread independiente de la interfaz del usuario.

La interfaz gráfica es uno de los aspectos importante a considerar para actualizaciones y mejoras futuras, ya que en esta iteración el cliente quería un grado complejo de funcionalidad en un tiempo muy corto por lo que se sacrificó tiempo de desarrollo de la interfaz por un mayor grado de funcionalidad.

4.4. Segunda Iteración

4.4.1. Planificación

Iteración 2	
Descripción	Desarrollo de la aplicación Emisor.
Historias de Usuario a Desarrollar	10.- Aplicación Emisor.
Tiempo Estimado	7 días

Tabla 29 Iteración 2

4.4.2. Tareas por Historia de Usuarios

HU 10.- Aplicación Emisor.

- Permite al usuario configurar aspectos relevantes al proceso de selección de datos del sistema administrativo Profit 2k8 (SQL Server) con el portal de integración (MySQL).
- Establecer la recurrencia de las actualizaciones en el portal de integración.
- Especificar los almacenes que estarán disponibles para su integración.
- Establecer las exclusiones de vendedores para evitar el envío de datos innecesarios al portal de integración.

4.4.3. Diseño

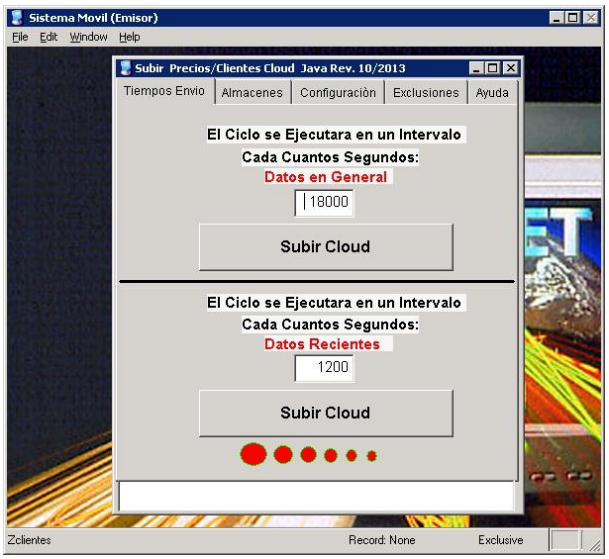


Figura 18. Diseño de la Aplicación Emisor

4.4.4. Codificación

Entre los aspectos más importantes se encuentran los parámetros de configuración que representan las opciones de filtrado. Lo cual mejora el desempeño de las sincronizaciones.

Figura 19. Formulario Configuración de la Aplicación Receptor

Objeto: Command1 Procedimiento:Click

```
xodbc=ALLTRIM(thisform.text5.Value)
xempresa=ALLTRIM(thisform.text1.Value)
xregistros=0
zstock=thisform.check9.Value
SET DATE BRITISH
wruta=ALLTRIM(thisform.text2.Value)
xp1 = SQLCONNECT(xodbc)
xprofit = SQLCONNECT('origen')
xt1="select * "
xt2="from pedidos where empresa ='"+xempresa + "'"
xsql1=xt1+xt2

** tomar los pedidos que hay en la web en mi cursor
       SQLEXEC(xp1,xsql1,'zpedidos')
** borrar los registros con cantidad=0
       xqueryborrar="delete from pedidos where cantidad=0 "
       SQLEXEC(xp1,xqueryborrar,'zborrar')
       SELECT zpedidos
       GO top
** agrupamos los pedidos
```

```
        SELECT INT(VAL(contador)) as contador1,co_art,empresa FROM zpedidos
        WHERE INT(VAL(contador)) >0 INTO CURSOR wpedidos1

        SELECT contador1 FROM wpedidos1 GROUP BY 1 ORDER BY 1 INTO CURSOR
        wpedidos
        SELECT * FROM wpedidos1 INTO CURSOR wpedidos9
… (abreviado)
*** desconecto el odbc de mysql
        SQLDISCONNECT(xp1)
IF xregistros>0 then
        thisform.label3.Caption="Subiendo al sistema " + STR(xregistros) +
        " Pedidos"
endif
** ahora proceso esos pedidos en profit plus

…...(abreviado)

**grabandolo en cotizaciones de clientes
SELECT wpedidos
GO top
DO WHILE NOT EOF()
        var1=contador1
        ……
        SELECT renglones
        GO top
        DO WHILE NOT EOF()
                var10=co_art
                var11=co_alma
                var12=cantidad
                var13=precio
                var14=unidad
                var99=ALLTRIM(comentario)
                xcli=co_cli
                xven=ALLTRIM(co_ven)
                xlen=LEN(xven)
                xxven=SUBSTR(xven,1,xlen)
                xclides=cli_des
                xdirec=nuevo_direc
                xrif=xcli
                tca0="insert into " + xempresa + ".dbo.cotiz_c
                (co_ven,fact_num,descrip,saldo,fe_us_in,fe_us_mo,fe_us_el,fec_emis,fec_venc,f
                eccom,co_cli,forma_pag,tot_bruto,tot_neto,"
                tca1="iva,tasa,moneda,co_sucu,co_tran,comentario,status,contr
                ib,campo1)"
                tca2=" values ('" + xxven + "','" + ALLTRIM(str(xnronuevo)) + "','Movil " +
                ALLTRIM(zcadena) + "','" + STR(xneto,14,2) + "','" + DTOS(var2) + "',"
                tca3="'"+DTOS(var2) + "','"+DTOS(var2) + "','" + DTOS(var2) + "','"
                +DTOS(var2)+ "','" + DTOS(var2) + "',"
                tca4= "'"+ALLTRIM(xcli) + "','" + ALLTRIM(xforma) + "','" +
                STR(xtotbruto,14,2) + "' ,'" + STR(xneto,14,2) + "','" + STR(xtotiva,14,2) +
                "','1','" + xmoneda + "','" + xsucursal + "','" + xtranspor + "','" + var99 +
                "',0,1,'" + xcli + "')"
                _cliptext=tca0+tca1+tca2+tca3+tca4
                SQLEXEC(xprofit,tca1+tca2+tca3+tca4,'zgrabar2')
        ENDIF
** tomar los renglones de este cliente
        xencabeza=0
        xrenglon=1
        SELECT renglones
        GO top
        DO WHILE NOT EOF()
                var10=co_art
                var11=co_alma
                var12=cantidad
                var13=precio
                var14=unidad
….
```

```
** agregar renglon
              ttca1="select * from " + xempresa + ".dbo.reng_cac where fact_num=?xnronuevo
              "
              SQLEXEC(xprofit,ttca1,'zbuscar10')
              SELECT zbuscar10
              GO top
              IF EOF()
                     xrenglon=1
              else
                     xrenglon=RECCOUNT()+1
              ENDIF
              tca1="select * from " + xempresa + ".dbo.reng_cac where co_art=?var10 and
              fact_num=?xnronuevo "
              SQLEXEC(xprofit,tca1,'zbuscar1')
              SELECT zbuscar1
              GO top
              IF EOF()
                     IF zstock=1 then
** buscar el articulo si la cantidad existe en el almacen
                     xquerystock="select co_art from " + xempresa + ".dbo.st_almac where
                     co_art=?var10 and co_alma=?var11 and  stock_act  >=?var12"
                     SQLEXEC(xprofit,xquerystock,'zstock')
                     SELECT zstock
                     GO top
                     IF EOF()
                            xsw=0
                     ELSE
                            xsw=1
                     ENDIF
                     *
                     IF xsw=1 then
                            tw1="insert into " + xempresa + ".dbo.reng_cac
                            (fact_num,reng_num,co_art,co_alma,total_art,pendiente,uni_vent
                            a,prec_vta,prec_vta2,tipo_imp,reng_neto,fec_lote)"
                            tw2="values ('" + str(xnronuevo) +"','" + STR(xrenglon) +
                            "','" + var10 + "','"+ var11 + "','" + STR(var12,14,2) + "','"
                            + STR(var12,14,2) + "','" + var14 + "','" + STR(var13,14,2) +
                            "','" + STR(var13,14,2) + "','" + var20 + "','" +
                            STR(xrengneto,14,2) + "','" + DTOS(var2)+ "')"
                            SQLEXEC(xprofit,tw1+tw2,'zgrabar1')
                            xsw=0
                     ENDIF
              ELSE
                     tw1="insert into " + xempresa + ".dbo.reng_cac
                     (fact_num,reng_num,co_art,co_alma,total_art,pendiente,uni_venta,prec_
                     vta,prec_vta2,tipo_imp,reng_neto,fec_lote)"
                     tw2="values ('" + str(xnronuevo) +"','" + STR(xrenglon) + "','" +
                     var10 + "','"+ var11 + "','" + STR(var12,14,2) + "','" +
                     STR(var12,14,2) + "','" + var14 + "','" + STR(var13,14,2) + "','" +
                     STR(var13,14,2) + "','" + var20 + "','" + STR(xrengneto,14,2) + "','"
                     + DTOS(var2)+ "')"
                     SQLEXEC(xprofit,tw1+tw2,'zgrabar1')

              ENDIF
       ENDIF
              SELECT renglones
              SKIP
              loop
       ENDDO
       SELECT wpedidos
       SKIP
       LOOP
ENDDO
** desconecto el odbc de sql
SQLDISCONNECT(xprofit)
thisform.label3.Caption=SPACE(50)
```

4.4.5. Pruebas

Las pruebas realizadas solo consistían en mandar la información perteneciente al ODBC origen para seleccionar los datos de la base de datos SQL del servidor del sistema Administrativo Profit 2k8, para agregarlo a la base de datos Mysql.

4.5. Tercera Iteración

4.5.1. Planificación

Iteración 3	
Descripción	Desarrollo de la aplicación Receptor.
Historias de Usuario a Desarrollar	11.- Aplicación Receptor.
Tiempo Estimado	7 días

Tabla 30 Iteración 3

4.5.2. Tareas por Historia de Usuarios

HU 11.- Aplicación Receptor.

- Permite al usuario configurar aspectos relevantes al proceso de selección de datos del portal de de integración (MySQL) para ser enviados al sistema administrativo profit 2k8.
- Establecer la recurrencia de las actualizaciones en el sistema administrativo profit 2k8.
- Especificar el intervalo de tiempo para cada recepción.

4.5.3. Diseño

El diseño de la Aplicación Receptor fue desarrollado con la herramienta visual fox pro, a través de un solo formulario que integra el origen de base de datos del cliente ODBC del

portal de integración con el archivo de datos del sistema administrativo profit 2k8. Como se puede observar en la Figura 20.

Figura 20. Diseño de Formulario de la Aplicación Receptor

4.5.4. Codificación

Se presentan los tres segmentos de código referentes a los objetos timer, encargado de asignar el tiempo de actualización y el procedimiento click del objeto command1 encargado de generar los pedidos al sistema profit 2k8 y la conformación de la licencia.

Objeto: Timer1 Procedimiento: Timer

```
thisform.timer1.Reset
DO case
      ** 1 minuto = 60.000 milisegundos
      CASE thisform.check1.Value=1
            ** 2 minutos
            thisform.timer1.Interval=120000
      CASE thisform.check2.Value=1
            ** 3 minutos
            thisform.timer1.Interval=180000
…..(abreviado)
      OTHERWISE
            ** 5 minutos
            thisform.timer1.Interval=300000
ENDCASE
      thisform.command1.SetFocus
      KEYBOARD '{ENTER}'
```

Figura 21. Diseño de Formulario de Conformación de Licencia Aplicación Receptor

Objeto: Command1 Procedimiento: Click

```
winstala=thisform.text1.Value
wlicenci=thisform.text2.Value
wconforma=thisform.text3.Value
wclave=thisform.text4.Value
_var2=INT((VAL(SUBSTR(DTOC(WINSTALA),1,2)+SUBSTR(DTOC(WINSTALA),4,2)+SUBS
TR(DTOC(WINSTALA),9,2))+VAL(WLICENCI)+Wconforma)*1.017)
IF wclave=_var2 then
      loFSO = CREATEOBJECT("Scripting.FileSystemObject")
      xunidad='"'+SYS(5)+'"'
      Xcadena="lofso.drives(" + xunidad + ").serialnumber"
      XnroSerial=&xcadena
      IF XnroSerial >0 then
      ELSE
            XnroSerial=XnroSerial*-1
      ENDIF

      xw1=INT(xnroserial*1.017)
      UPDATE ICONFIG.JC SET serial=xw1
      CLOSE DATABASES
      DO FORM ftpreceptor
ELSE
      thisform.label4.Visible= .T.
      thisform.label4.Caption="Error de Conformaciòn!!!!"
      WAIT "" TIMEOUT 2
      CLOSE ALL
      CLEAR
      cancel
endif
```

4.5.5. Pruebas

Las pruebas de usuario fueron basadas en los requerimientos propios de la Automatización del sistema por consiguiente la única interacción con el usuario es para la definición de los parámetros de configuración y el intervalo de ejecución. Solamente se verifico que cargara la información correctamente al sistema administrativo profit 2k8.

4.6. Cuarta Iteración

4.6.1. Planificación

Iteración 4	
Descripción	Desarrollo del Portal de Integración.
Historias de Usuario a Desarrollar	12.- Portal de Integración.
Tiempo Estimado	7 días

Tabla 31 Iteración 4

4.6.2. Tareas por Historia de Usuarios

HU 12.- Portal de Integración.

- Instalación del servidor web mediante la instalación del paquete wamp que contiene todos elementos necesarios para poder llevar a cabo nuestro portal de integración http://wampserver.com.
- Definir la estructura para la base de datos MySql.
- Crear los códigos Php para intercambio de datos con la Aplicación Móvil.

4.6.3. Diseño

El diseño del portal de integración consta de dos partes esenciales: La base de datos definida en MySql y denominada “movil”, mediante la herramienta phpMyAdmin.

En la base de datos movil, se generaron las tablas necesarias para modelar el proceso de automatización de la fuerza de ventas.

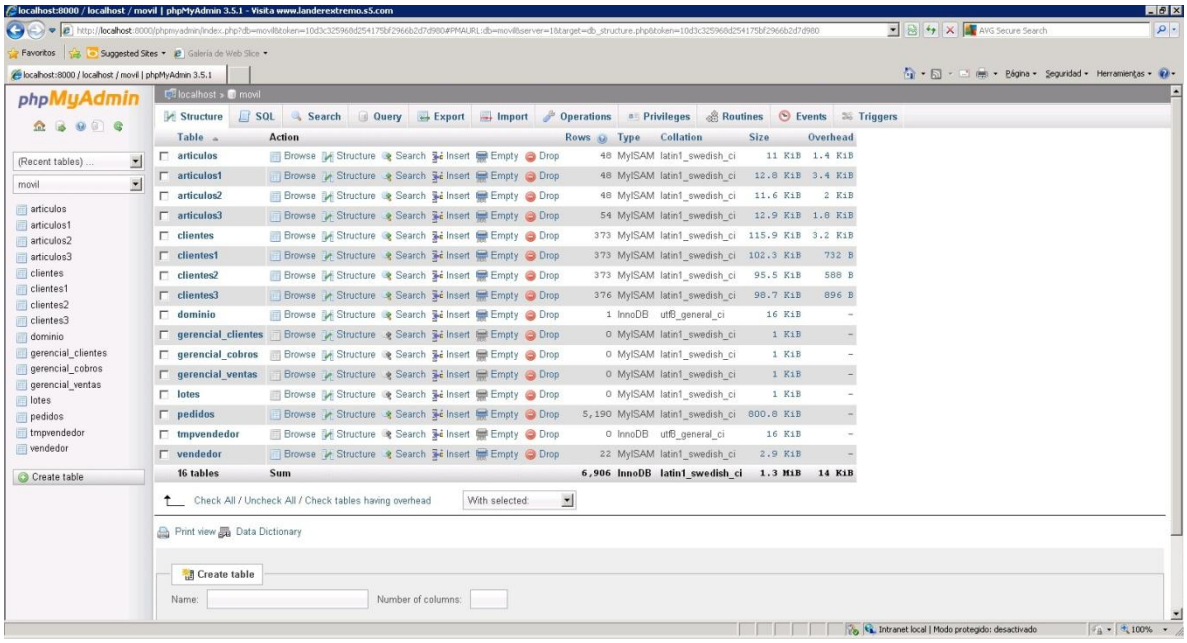


Figura 22. Diseño de la base de datos “movil”

4.6.4. Codificación

La codificación del portal de integración contiene el código Php necesario sólo para la interacción con la Aplicación Móvil, ya que para la Aplicación Emisor y la Aplicación Receptor únicamente se accede a la base de datos.

A continuación se presentan los códigos más relevantes para el proceso de sincronización y recepción de los pedidos.

Conexion.php

```
<?php
$conexion= mysql_connect("localhost","root","") or die("error de conexión
al servidor");
mysql_select_db("movil",$conexion) or die("error de conexión a la BD");
?>
```

AllDataArt.php

```
<?php
include "conexion.php";
$sqlart=mysql_query("select * from articulos");
while($e=mysql_fetch_assoc($sqlart))
       $output[]=$e;
              print(json_encode($output));
        mysql_close($conexion);
?>
```

AllDataCli.php

```
<?php
include "conexion.php";
// extraer el codigo de vendedor temporal y luego limpiar dicha tabla
$extraerven=mysql_query("select coven from tmpvendedor");
while($ven=mysql_fetch_assoc($extraerven))
       $vendedor1=$ven['coven'];
// vaciar la tabla tmpvendedor
$vaciar=mysql_query("delete from tmpvendedor");
$sqlcli=mysql_query("select  from clientes where co_ven='".$vendedor1."'");
while($e=mysql_fetch_assoc($sqlcli))
       $output[]=$e;
              print(json_encode($output));
mysql_close($conexion);
?>
```

Subirpd1.php

```
<?php
include "conexion.php";
$data = json_decode( file_get_contents('php://input') );
$filas=count($data);
$respuesta=1;
             $x1= $data->campo1;  // codigo articulo
             $x2= $data->campo2;  // nombre articulo
             $x3= $data->campo3;  // cantidad
             $x4= $data->campo4;  // codigo almacen
             $x5= $data->campo5;  // precio
             $x6= $data->campo6;  // unidad
             $x7= $data->campo7;  // codigo cliente
             $x8= $data->campo8;  // nombre cliente
             $x9= $data->campo9;  // codigo vendedor
             $x10=  date('Y-m-d');       // fecha del pedido
             $x11= $data->campo11;       // comentario del pedido
             $x12= $data->campo12;       // correlativo o nro del pedido
             $x13= $data->campo13;       // codigo empresa
             $x14= $data->campo14;       // direccion de cliente nuevo
$sql2="insert into pedidos
(co_art,art_des,co_alma,unidad,cantidad,precio,co_cli,cli_des,co_ven,fecha,coment
ario,contador,empresa,nuevo_direc)
values('".$x1."','".$x2."','".$x4."','".$x6."','".$x3."','".$x5."','".$x7."','".$
x8."','".$x9."','".$x10."','".$x11."','".$x12."','".$x13."','".$x14."')";

       $consulta1= mysql_query($sql2,$conexion);
       // cerrar la conexion en el servidor
        mysql_close($conexion);
       print_r(json_encode($respuesta));
?>
```

4.6.5. Pruebas

Las pruebas realizadas al portal de integración consistieron en verificar el acceso a la base de datos, y la verificación de los códigos Php.

Uno de los aspectos que debimos tomar en consideración para la fase de pruebas fue definir un nombre de dominio ya que la dirección de acceso pública asignada a la red era dinámica y cambiaba con frecuencia.

Para solventar el problema se creó un dns gratuito a través del portal www.no-ip.com con el nombre de galicia300ca.ddns.net. Adicionalmente direccionó el puerto utilizado por php para el acceso a través de la aplicación móvil mediante la función NAT del Router de la empresa. En nuestro caso se tomó el puerto 8000.

De manera tal que el acceso al portal web desde nuestra aplicación móvil fue definido de la siguiente manera: galicia300ca.ddns.net:8000.

Conclusiones

Una vez concluido el presente T.E.G podemos aseverar que hemos alcanzado los objetivos propuestos con el mismo, puesto que por un lado partimos de una problemática a solucionar, la cual consistía en: automatizar el proceso de la fuerza de ventas de una compañía del ramo de alimentos, específicamente de la Distribuidora de Quesos Galicia 300, C.A.

El T.E.G. me permitió comprender y determinar que una Aplicación Móvil por sí sola no podría solucionar por completo la automatización del proceso de elaboración de los pedidos realizados por la fuerza de ventas, para resolver tal situación fue necesario crear una arquitectura que permitiera a la Aplicación Móvil integrar toda su potencialidad con el sistema Administrativo Profit 2K8 disponible en la empresa, mediante la elaboración de tres componentes adicionales: una Aplicación Emisor, una Aplicación Receptor y un Portal de Integración.

La personalización XP para nuestro proyecto fue clave para el éxito del mismo, gracias a la aplicación de prácticas del proceso en diseño (pudiendo incluir parte de Modelación Ágil), codificación y pruebas que soportaron el desarrollo rápido de una aplicación expuesta a cambios durante las etapas de su creación, comprometido en cumplir con cada iteración. Además, trabajar con los factores de prioridad, riesgo y tiempo determinó la toma de decisiones al momento de planificar que crearon un buen ritmo de implementación y solución de requerimientos funcionales y no funcionales en el momento indicado.

Por otra parte, es la primera vez que me planteaba desarrollar una Aplicación Móvil, esto me llevó a explorar ejemplos, aprender la concepción del proceso de pedidos utilizado por la fuerza de ventas para su automatización, investigar sobre los conceptos que manejan, su funcionamiento y resultados, lo cual implica un aprendizaje importante sobre las información utilizada para manejar la interacción de los usuarios con esta aplicación y como poder integrarla con una plataforma diferente.

Finalmente, una gran experiencia fue haber aprendido y aplicado en este proyecto la combinación de varios Lenguajes y Herramientas de Programación tales como: Java, Php, Xml, Visual FoxPro y Eclipse, así como varios sistemas manejadores de bases de datos: Mysql, SQLite y SQL Server, para modelar el intercambio de la información contenida entre el Sistema Administrativo Profit 2K8, el cual contiene su data en SQL Server, la Aplicación móvil con SQLite, y el almacenaje de datos en el portal de integración bajo Mysql.

Glosario

Eclipse: Es una plataforma de software compuesto por un conjunto de herramientas de programación de código abierto multiplataforma, típicamente ha sido usada para desarrollar entornos de desarrollo integrados (del inglés IDE), como el IDE de Java llamado Java Development Toolkit (JDT).

DNS: Domain Name System o DNS en español (Sistema de Nombres de Dominio), es un sistema de nomenclatura jerárquico descentralizado para dispositivos conectados a redes IP como Internet o una red privada.

Framework: Es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos concretos de software, que puede servir de base para la organización y desarrollo de software.

Intent: Es una declaración de necesidades. Un objeto Intent está conformado por fragmentos de información que describen la acción o el servicio deseado.

IntentFilter: Es una declaración de capacidad e interés por ofrecer asistencia a los que la necesitan.

Java: Es un lenguaje de programación de propósito general, concurrente, orientado a objetos que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible.

MySQL: Es un sistema de gestión de bases de datos relacional desarrollado bajo licencia dual GPL/Licencia comercial por Oracle Corporation y está considerada como la base datos open source más popular del mundo.

NAT: La traducción de direcciones de red o NAT (del inglés Network Address Translation) es un mecanismo utilizado por routers IP para intercambiar paquetes entre dos redes que asignan mutuamente direcciones incompatibles. Consiste en convertir, en tiempo real, las direcciones utilizadas en los paquetes transportados.

ODBC: Open DataBase Connectivity (ODBC) es un estándar de acceso a las bases de datos desarrollado por SQL Access Group (SAG) en 1992. El objetivo de ODBC es hacer posible el acceder a cualquier dato desde cualquier aplicación, sin importar qué sistema de gestión de bases de datos (DBMS) almacene los datos.

Php: Es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico.

phpMyAdmin: Es una herramienta escrita en PHP con la intención de manejar la administración de MySQL a través de páginas web, utilizando Internet. Actualmente puede crear y eliminar Bases de Datos, crear, eliminar y alterar tablas, borrar, editar y añadir campos, ejecutar cualquier sentencia SQL, administrar claves en campos, administrar privilegios.

Plugin: Es un programa de ordenador que interactúa con otro programa para aportarle una función o utilidad específica.

Servidor: Es un ordenador o software que realiza alguna tarea como servicio de otras aplicaciones llamadas clientes.

SQL Server: Es un sistema de manejo de bases de datos del modelo relacional, desarrollado por la empresa Microsoft.

SQLite: Es un sistema de gestión de bases de datos relacional compatible con ACID, contenida en una relativamente pequeña biblioteca escrita en C.

Xml: Siglas en inglés de eXtensible Markup Language ("lenguaje de marcas Extensible"), es un lenguaje de marcas desarrollado por el World Wide Web Consortium(W3C) utilizado para almacenar datos en forma legible.

Wamp: Es el acrónimo usado para describir un sistema de infraestructura de internet que usa las siguientes herramientas:

- Windows, como sistema operativo;
- Apache, como servidor web;
- MySQL, como gestor de bases de datos;
- PHP (generalmente), Perl, o Python, como lenguajes de programación.

Referencias

[1]. CeluActivo, Introducción a la programación en Android, fecha de recuperación 17 de noviembre de 2012, http://www.celuactivo.com/2012/08/27/introduccion-a-la-programacion-en-android/. 2012.

[2]. Dosisgeek, Blog Geek, Razones para usar Android, Recuperado de: http://www.tudosisgeek.com/10-razones-para-usar-android-ventajas/. 2015.

[3]. Distribución de Android según su versión. Información desde la versión 2.3.3 hasta la versión para Mayo 2016. Fecha de recuperación 11 de Julio de 2016: https://developer.android.com/about/dashboards/index.html. 2016.

[4]. Evolución de ventas de dispositivos según su Sistema Operativo. Fecha a partir del año 2007 hasta el 2013. Fecha de recuperación 06 de Junio de 2016: https://en.wikipedia.org/wiki/Mobile_operating_system#Market_Share_by_Country_or_Region. 2016.

[5]. Arquitectura de componentes que conforman Android. Recuperado de: http://developer.android.com/guide/basics/what-is-android.html. 2016.

[6]. Difference of DVM and JVM in the Android World. Recuperado de: http://www.momob.in/2010/general/bhavis/difference-of-dvm-and-jvm-in-the-android-world. 2001.

[7]. Ableson W. Frank, Sen Robi, King Chris “Android Guía para desarrolladores”, Anaya Multimedia, Segunda Edición, 2011.

[8]. Beck K., Beedle M., Van Bennekum A., Cockburn A., Cunningham W., F. & M., Thomas D. Manifiesto por el Desarrollo Ágil de Software, Recuperado de: http://agilemanifiesto.org/iso/es/manifiesto.html.

[9]. Aguilar Sierra, Alejandro, “Introducción a la Programación Extrema”, Universidad Nacional Autónoma de México, Diciembre, 2002.

[10]. Beck Kent, Martin Fowler, “Planning Extreme Programming”, Addison Wesley, Primera Edicion, Octubre 2000.

[11]. Beck, K., Extreme Programming Explained. Embrace Change. Traducido al español como: Una explicación de la programación extrema. Aceptar el cambio, Person Education, 2000.