UNIVERSIDAD SIMÓN BOLÍVAR
DECANATO DE ESTUDIOS PROFESIONALES
COORDINACIÓN DE INGENIERÍA DE LA COMPUTACIÓN

IMPLEMENTACIÓN DE UN COMPONENTE PUBLICITARIO (CLIENTE) PARA APLICACIONES MÓVILES

Por:
Sabrina María Fernández Mirás

INFORME DE PASANTÍA

Presentado ante la Ilustre Universidad Simón Bolívar
como requisito parcial para optar por el título de
Ingeniero en Computación

Sartenejas, Septiembre de 2012

UNIVERSIDAD SIMÓN BOLÍVAR
DECANATO DE ESTUDIOS PROFESIONALES
COORDINACIÓN DE INGENIERÍA DE LA COMPUTACIÓN

IMPLEMENTACIÓN DE UN COMPONENTE PUBLICITARIO (CLIENTE) PARA APLICACIONES MÓVILES

Por:
Sabrina María Fernández Mirás

Realizado con la asesoría de:
Tutor Académico: Prof. Ángela Di Serio
Tutor Industrial: Ing. Carlos Marcano

INFORME DE PASANTÍA

Presentado ante la Ilustre Universidad Simón Bolívar
como requisito parcial para optar por el título de
Ingeniero en Computación

Sartenejas, Septiembre de 2012

UNIVERSIDAD SIMÓN BOLÍVAR
DECANATO DE ESTUDIOS PROFESIONALES
COORDINACIÓN DE INGENIERÍA DE LA COMPUTACIÓN

ACTA FINAL PASANTÍA LARGA

IMPLEMENTACIÓN DE UN COMPONENTE PUBLICITARIO (CLIENTE) PARA APLICACIONES MÓVILES

Presentado por:
Sabrina María Fernández Mirás

Esta Pasantía Larga ha sido aprobada por el siguiente jurado examinador:

Prof. _____
Jurado

Prof. Ángela Di Serio
Tutor Académico

Ing. Carlos Marcano
Tutor Industrial

Sartenejas, 01/09/2012

Resumen

El presente proyecto de pasantía consiste en el diseño y la implementación de un componente publicitario para ser incluído en las aplicaciones desarrolladas para los sistemas operativos móviles iOS y Android. La empresa Mobile Media Networks C.A. tiene como objetivo ampliar su alcance publicitario correspondiente a la colocacion de campañas en aplicaciones móviles. Gracias al éxito de ventas de campañas publicitarias con la plataforma Blackberry, se desea llegar a los usuarios de otras tecnologías que están siendo cada vez más populares en Venezuela. Estas son iOS por la empresa Apple y Android de la empresa Google. Por lo tanto es necesario la creación de un componente nativo por cada plataforma que permita incluir de manera sencilla la publicidad en las aplicaciones.

El componente actual, creado para aplicaciones de Blackberry, se comunica con el servidor mediante un protocolo que genera una respuesta muy pesada con información inutil para la colocación de la piblicidad. Por lo tanto el primer paso es el diseño de un nuevo protocolo que produzca una respuesta de menor tamaño y con la información mínima necesaria, lo que tendrá un impacto favorable tanto en espacio como en tiempo. Una vez tomada la decisión sobre la comunicación con el servidor, se realizará el diseño de los componentes, analizando el actual y considerando las posibles mejoras. El componente trabaja en conjunto con el AdServer MobileX, un servidor que se encarga de enviar la publicidad de manera inteligente. La realización de los cambios en el servidor para implementar la nueva comunicación no está en el alcance de este proyecto.

El componente debe ser capaz de colocar en cualquier lugar de la pantalla del dispositivo la publicidad, sin importar el tipo de la imagen o su tamaño original. Además debe reportar al servidor cada vez que se mostró o que el usuario hizo clic sobre ella. La entrega oportuna de esta información es crucial para la venta apropiada de campañas a los clientes y la generación de reportes de las mismas. Finalmente, debe manejar problemas como fallas en la conexión, el almacenamiento o la posible sobrecarga del servidor, así como algunas necesidades nuevas que permiten segmentar la publicidad por país o por tipo de usuario.

La metodología utilizada para la implementación de este proyecto es AUP, una metodología de tipo ágil, con un buen balance entre la planificación del proyecto y tiempos de desarrollo del software.

DEDICATORIA

A mis padres,
por su apoyo incondicional y su amor.
Gracias a ustedes he cumplido mis metas.
A mis hermanas,
por estar siempre conmigo
y ser mis amigas más queridas.

Agradecimientos

Gracias a Mobile Media Networks C.A. por brindarme esta oportunidad única. Ser escogida para formar parte de un equipo de trabajo tan importante fue un honor para mí. En especial a los Ings. Miguel Sucre y Ricardo Blanch, su visión de futuro y emprendimiento marcaron la diferencia. También a mis compañeros de trabajo María Gabriela, Alechandrina, Ariana, Ricardo, Verónica, Yessenia, Cesar, Ciuffi, Alicia, Andrés, Edgardo, José Eduardo y al resto de las personas que forman parte del equipo de Mobile Media, quienes me brindaron un ambiente de trabajo ameno, divertido y lleno de conocimientos.

A mi tutor industrial, el Ing. Carlos Marcano, por su paciencia y apoyo incondicional. Sus consejos y ayuda hicieron posible que el proyecto cumpliera las expectativas. A mi tutora académica, la Profesora Ángela Di Serio, por su asesoría en el desarrollo del proyecto, siempre buscando que use todos los conocimientos aprendidos a lo largo de mi carrera. Nunca dudó que lograría cumplir con todos los objetivos.

A mis compañeros de carrera Miguel, Daniel, Manuel, Christian, Daniela, Fabio, Ricardo, María Gracia, y Fernando, con los cuales compartí buenos y malos momentos. A Mariana Bezada, que tanto tiene que enseñarme todos los días, siempre la primera en ayudarme. A Mayerlin González, por hacer de la Universidad una experiencia divertida. A Jonathan, porque aunque nos separe todo un océano siempre estás conmigo para asesorarme, nunca has dejado de estar pendiente de mí. Y no podía faltar Mariana Rodríguez, por ser mi tercera hermana. Dieciocho años de amistad han sido pocos con respecto a los que nos faltan.

A mis padres, María Jesús y Arturo, por no dejarme olvidar mi meta. Han sido mi apoyo y mi mayor fortaleza. Gracias por su amor incondicional a pesar de mi difícil carácter. A mi hermana Cristina, quien a pesar de ser menor que yo todos los días tiene una lección de madurez que darme. A mi hermana Mónica, por siempre tener una sonrisa para mi. A mi abuela María Dolores, y a todos los que no he mencionado aquí, que aunque no estén lejos, siempre los tengo cerca de mí corazón.

A la Universidad Simón Bolívar, por convertirse en mi segunda casa.

A Dios, por darme fuerzas y esperanza.

A todos ustedes, mil gracias.

Índice general

Índice de cuadros

Índice de figuras

Lista de Símbolos y Abreviaturas

API	Application Programming Interface
AUP	Agile Unified Process
HTTP	Hypertext Transfer Protocol
IDE	Integrated Development Enviroment
IP	Internet Protocol
JSON	JavaScript Object Notation
MCC	Mobile Country Code
MNC	Mobile Network Code
RPC	Remote Procedure Call
RUP	Rational Unified Process
XML	eXtensible Markup Language

Introducción

El mundo de la tecnología cambia día a día, donde los usuarios experimentan continuamente con una gran cantidad de formas de tecnologías creativas e innovadoras. Cada semana salen al mercado aplicaciones móviles o portales web que ofrecen servicios que hacen más fácil y entretenida la vida de las personas, tales como las tiendas y bancos virtuales, el correo electrónico y los mensajes instantáneos.

Es así que el consumo crece en igual medida, por lo que comunicar en masa a través de este medio es de gran importancia para las empresas que deseen dar a conocer sus productos o servicios.

La colocación de publicidad en los dispositivos móviles es un suceso reciente que ha dado buenos resultados. Mobile Media Networks se especializa en esta área, siendo uno de sus servicios ofrecidos la distribución de publicidad en aplicaciones de dispositivos móviles. Esta distribución consiste en la colocación de la publicidad y en la presentación de los reportes a los anunciantes. El reporte consta de datos importantes, como impresiones, interacciones y el porcentaje de efectividad. Sin embargo los métodos de mercado se hacen cada día más exigentes y ambiciosos con la idea de promover acciones en los usuarios de la publicidad. Los anunciantes desean dirigir su publicidad a quién muestre interés por ella, a quien le sea relevante.

La empresa utiliza para la distribución de publicidad en los dispositivos móviles, el componente AdField MobileX de Blackberry, en conjunto con el sistema AdServer MobileX, basado en la plataforma OpenX. En base a las necesidades se ha planteado la creación de los componentes publicitarios para las plataformas iOS y Android, incluyendo nuevas funcionalidades que el AdField no contempla actualmente.

El presente proyecto describe el proceso de desarrollo del componente publicitario AdView MobileX, que permita la colocación de las creatividades en las aplicaciones móviles, así como el reporte de las impresiones y clics realizados, el envío de datos importantes de segmentación al servidor, y el almacenamiento y rotación de las creatividades. Fue desarrollado bajo la metodología AUP en un período de 20 semanas.

El informe desarrollado para este proyecto de pasantía larga presenta una estructura en capítulos. El primer capítulo describe el entorno laboral, para la ubicación del escenario del problema desarrollado. El segundo capítulo comprende el marco teórico donde se exponen los conceptos necesarios para la comprensión del problema y de la solución realizada. El tercer capítulo muestra la metodología utilizada AUP. El cuarto capítulo presenta el marco tecnológico describiendo las herramientas utilizadas para el desarrollo del proyecto. En el

capítulo quinto se describe el desarrollo del proyecto y los resultados basado en la metodología usada. Finalmente se plantean las conclusiones y recomendaciones.

A continuación, se exponen los objetivos generales y específicos que se busca alcanzar en este desarrollo, y el alcance que abarca este proyecto. Esta información tiene la intención de que el lector comprenda el contexto general del proyecto y conozca el propósito del mismo.

Objetivo general

Diseñar un protocolo eficiente y un componente nativo para cada plataforma (IOS y Android) que permita entregar y reportar los impactos publicitarios asignados.

Objetivos específicos

- Diseñar un protocolo que permita enviar toda la información necesaria para mostrar la publicidad y reportar de vuelta la correcta entrega de los datos y las acciones tomadas por el usuario.
- Diseñar un componente que se acople con el protocolo planteado y permita la recepción y visualización de la información enviada por el AdServerMobileX.
- Desarrollo de un componente nativo IOS a partir del diseño propuesto.
- Desarrollo de un componente nativo Android a partir del diseño propuesto.

Alcance

El alcance de este proyecto es el desarrollo de librerías para la introducción de publicidad interactiva en las aplicaciones nativas con el objetivo de ofrecer una presentación personalizada. La entrega del prototipo será 100 % funcional.

Capítulo 1

Entorno Empresarial

Este capítulo da a conocer el entorno empresarial en el cual se desarrolla el proyecto de pasantías. Busca mostrar la labor de Mobile Media Networks, dando una idea de sus objetivos, misión y aspectos históricos relevantes.

Mobile Media Networks

La empresa Mobile Media Networks se dedica al desarrollo de portales y aplicaciones en dispositivos móviles, así como la distribución de publicidad en los mismos. A través del manejo de distintas plataformas móviles, diseñan y programan aplicaciones que logran integrar el contenido y la publicidad de distintas empresas a tabletas, celulares y otros dispositivos portatiles. A continuación se presentará una breve descripción de la empresa.

Reseña Histórica

Mobile Media Networks nace de la empresa "Adsmedia Mobile Advertising" de España, la cual se dedica al mercadeo móvil y a la publicidad. Es en Barcelona, en el año 2006, cuando es fundada Adsmedia Mobile Advertising, como una empresa global, joven y dinámica con la capacidad de adaptar de forma inmediata la oferta de productos y servicios publicitarios a las necesidades nuevas y cambiantes del mercado.

En el año 2007 gana el premio "Empresa Innovadora" de la Generalitat de Catalunya, el premio "las 100 mejores ideas empresariales" otorgado por la revista Actualidad Económica y queda como finalista en BCN-Emprendedores edición 2007. Al poco tiempo es seleccionada por Nokia como "partner" estratégico para el desarrollo de Publicidad Móvil en España, Portugal y Latinoamérica. En mayo de este año se abren nuevas oficinas en Londres y en Caracas.[1]

En el 2008 surgen nuevos contratos de gestión y desarrollo con empresas como Movistar, Vodafone, Orange, Meridiano y Digitel. También tienen lugar expansiones y nuevas versiones del sistema encargado de la distribución de publicidad, "Mobile Advertising Adserver", ofreciendo nuevas tecnologías como el soporte de videos. También se realiza la apertura de oficinas en Brasil.

En el año siguiente es desarrollada y puesta en uso la nueva versión del sistema "Mobile

Advertising Adserver" y nuevas oficinas son inauguradas en Bahrein y Lisboa. Por otro lado la empresa tiene presencia en el Mobile World Congress 2009. [1]

A mediados del año 2009 ocurre el traspaso de las operaciones de "Adsmedia Mobile Advertising" en España a "Adsmedia Mobile Advertising" en Venezuela, otorgando a estas oficinas la gestión del conjunto de sucursales distribuidas a nivel mundial y el control de su desarrollo. Poco después esta empresa adquiere independencia con respecto a la sucursal en España y en el año 2010 cambian su nombre a "Mobile Media Networks".

Actualmente Mobile Media Networks posee oficinas en Caracas, Bogotá y Miami, prestando los servicios de distribución de publicidad en portales web y dispositivos portátiles, así como el desarrollo de aplicaciones móviles para las plataformas iOS, Android y Blackberry.

Misión

Mobile Media Networks es la empresa especializada en brindar soluciones de mercadeo en celulares y publicidad en Latinoamérica y la ciudad de Miami.

Visión

Mobile Media Networks presenta su visión como:

"Creer en la liberalización de las comunicaciones y en el desarrollo de la publicidad en el móvil. El móvil se ha convertido en un nuevo soporte publicitario, por ello en Mobile Media Networks se persigue generar valor para toda la cadena implicada, desde los usuarios hasta las agencias de medios, pasando por los operadores y los anunciantes.

Incorporar publicidad a toda la cadena de comunicación, con lo que se incrementa el acceso de los anunciantes a las comunicaciones, generando servicios gratuitos y, lo más importante, reduciendo los costes para el usuario."[2]

Estructura organizacional

En la Figura 1.1, se muestra la estructura organizacional de Mobile Media Networks.

.

La realización del proyecto de pasantía fue bajo el cargo de 'Pasante' en la dirección de Tecnología de Mobile Media Networks.

Resumen del Cargo de Pasante de Tecnología

El pasante de Tecnología es el estudiante que busca conocimientos que lo ayuden a desarrollarse como profesional. Las pasantías académicas son las que representan un requisito para graduarse y se caracteriza por tener tiempos académicos y proyectos específicos.[3]

Capítulo 2

Marco Teórico

A continuación, se presentan los conceptos teóricos relevantes sobre los cuales se fundamentó el proyecto de pasantía, que incluyen el marco de trabajo y las herramientas utilizadas.

Conceptos Básicos

Mobile X

Es una herramienta para la gestión de campañas publicitarias en dispositivos móviles basado en OpenX. MobileX cumple con el modelo cliente-servidor. El cliente es el componente encargado de mostrar la publicidad en un portal web o en una aplicación, mientras que el servidor es el responsable de proveer al cliente la publicidad adecuada, la cual se identifica mediante un número llamado identificador de zona (zoneId).

La comunicación entre ambos se realiza mediante el protocolo XML-RPC para la versión 1.0 del cliente, y el protocolo JSON-RPC para la versión 2.0.

AdServer MobileX

Es una plataforma publicitaria para la entrega de publicidad en dispositivos móviles, basado en el AdServer de OpenX. Posee características para el manejo y la distribución de campañas y un sistema de rastreo para presentar las estadísticas.

Está licenciado bajo GNU General Public License y escrito en su mayor parte en PHP, utilizando MySQL como manejador de base de datos. La instalación esta hecha sobre los servidores de Amazon, bajo el servicio web de computación de nube (Cloud Computing) Amazon EC2 (Elastic Cloud Computing).

El AdServer maneja varios módulos para la entrega y el reporte de las campañas. Una campaña es un conjunto específico de banners, los cuales tienen una fecha de inicio y de finalización, así como una frecuencia con la que serán mostrados. Se manejan tres tipos de campaña:

1. Exclusiva: es aquella campaña que se ejecuta antes que cualquier otra.

2. De contrato: es aquella campaña que será entregada un número específico de veces por días, por un número específico de días.

3. Remanente: es aquella campana que se mostrará luego de las anteriores, limitada por un número de impresiones o una fecha de finalización.

Un banner es cualquier tipo de contenido creativo que será mostrado como una publicidad. Puede ser de formato JPG, PNG, GIF y SWF, así como ser escrito en Javascript, HTML. Un banner para una campaña puede tener varias versiones por los distintos tamaños predeterminados de los dispositivos actuales para los cuales será distribuida la publicidad.

AdField MobileX

Es un componente publicitario que actúa como cliente para la plataforma Blackberry cuya finalidad es mostrar las campañas publicitarias en los dispositivos, así como reportar la impresión y el clic de las mismas. Esta escrito completamente en java, utilizando algunas librerías de RIM. Su nombre se debe a que en Blackberry un área rectangular se denomina un campo (field), por lo que una creatividad se adapta a esta definición.

Actualmente se encuentra operativa la versión 1.0 del componente y en desarrollo la versión 2.0 en base a las nuevas funcionalidades y cambios propuestos en este proyecto de pasantía.

Para poder desplegar el contenido publicitario, el cliente debe realizar una solicitud al servidor enviando un número de zona. La zona identifica un espacio donde las formas creativas son desplegadas. Cuando el servidor recibe el número de zona, ejecuta un código de invocación, el cual retorna la información del banner necesaria para ser mostrado.

La información mínima que se debe tener para poder desplegar el banner es:

1. Dirección de contenido: dirección url donde se encuentra la creatividad.

2. Dirección de impresión: dirección url que será llamada cuando se muestre la creatividad en la aplicación.

3. Dirección de clic: dirección url que será llamada cuando se haga clic sobre la creatividad en la aplicación.

4. Ancho: ancho que mide la creatividad. (Debe ser menor que el ancho de la pantalla del dispositivo).

5. Alto : alto que mide la creatividad. (Debe ser menor que el alto de la pantalla del dispositivo).

6. Identificador de banner: identificador del tipo de creatividad.
7. Tipo de contenido: define si la creatividad es de formato jpg, png, gif, entre otros posibles.
8. Tipo de campaña: define si la campaña es exclusiva, de contrato o remanente.

AdView MobileX

Es un componente publicitario que actúa como cliente para las plataformas iOS y Android. Este componente cumple con las mismas funcionalidades que el componente AdField MobileX, solo que adopta este nombre porque un área rectangular en estas plataformas se denomina una vista (view), término bajo el cual se adapta la definición de una creatividad.

La versión de Android está escrito completamente en Java, utilizando algunas librerías de propias de Android, mientras que la versión de iOS está escrito en Objective-C. Para ambas versiones fue necesaria la utilización de algunas librerías de terceros para implementar algunas funcionalidades.

Actualmente se encuentra operativa la versión 2.0 del componente, incluyendo los cambios pertinentes del servidor.

Para esta versión el cliente debe realizar una solicitud al servidor enviando un número de zona, el tamaño de la pantalla (ancho y alto), un identificador del registro (cookie), el Código de Area de País (Mobile country code) y el Código de Area del Operador de Telefonia (Mobile network code).

Herramientas

Modelo Cliente - Servidor

La arquitectura cliente-servidor permite al usuario en un dispositivo movil, llamado el cliente, requerir algún servicio de una máquina a la que está unido, llamado el servidor, mediante una red, ya sea de datos o inalamrbica (Wi-Fi). Estos servicios pueden ser peticiones de datos de una base de datos, de información contenida en archivos o los archivos en sí mismos, o peticiones de imprimir datos en una impresora asociada [4].

El modelo cliente-servidor se fundamenta en la idea de que un proveedor de datos por si mismo puede abastecer las solicitudes de numerosos clientes que requieren de sus servicios.

El componente es un ejemplo de un sistema basado en este modelo. Cada cliente estará

representado por el componente publicitario, que realizará solicitudes al servidor en el momento oportuno solicitando una creatividad; y el servidor, que contiene la aplicación de gestión de solicitudes de publicidad, atenderá cada una de las solicitudes del cliente.

Agente de Usuario

El agente de usuario es una cadena de caracteres que identifica a un dispositivo o un portal web. Los datos que incluye son:

1. El nombre de la aplicación y la versión.
2. El tipo de explorador y su versión.
3. El sistema operativo.
4. Las extensiones instaladas en el sistema o en el explorador.

Figura 2.1: Formato del agente de usuario [5]

Librerías de Terceros

Las librerías de terceros (Third-party software component) son componentes desarrollados para implementar alguna funcionalidad que no esté presente en la plataforma que se trabaja o para mejorar las que existan. Pueden ser vendidas o distribuidas libremente, por lo que cada librería posee una licencia asociada límita su uso. La ventaja de estas librerías es que el desarrollador no tiene que estar escribiendo un nuevo componente, pero tiene que estar conciente de que existe la posibilidad de que el componente presente errores que el creador original no manejó.

Capítulo 3

Marco Metodológico

Este capítulo presenta los aspectos metodológicos para el desarrollo del proyecto de pasantía, se realiza una breve descripción la metodología utilizada así como su implementación en este proyecto.

AUP

El Proceso Unificado Agil (*Agile Unified Process*, AUP o AgileUP) es una metodología para el desarrollo de software, basado en el Rational Unified Process (RUP) de IBM. Describe de una forma simple la forma de desarrollar aplicaciones de software de negocio utilizando técnicas ágiles y conceptos que aún se mantienen válidos en RUP. [7]

AUP emplea técnicas ágiles, incluyendo Desarrollo Guiado por Pruebas (*Test Driven Development* o TDD), modelado ágil, gestión ágil de cambios y refactorización de bases de datos. Algunas de estas técnicas formaron parte integral del desarrollo del proyecto.

AUP establece metas y objetivos donde las colaboraciones son usadas para lograr resultados. Una colaboración puede referirse a un trabajo realizado por un sólo individuo que interactúa con otros, o puede referirse a un grupo de trabajo que involucra a múltiples individuos que interactúan unos con otros y con agentes externos. [6]

Para el desarrollo de este proyecto la metodología AUP fue implementada siguiendo la documentación de la versión 1.1 del Proceso Unificado Ágil. Se emplea AUP debido a que presenta un equilibrio entre las etapas de diseño, planificación y documentación del sistema, con la implementación del mismo, por lo tanto, se sigue manteniendo un desarrollo estructurado y coherente, pero brindando una mayor flexibilidad para el programador, tanto para en tiempo efectivo de implementación, como en el manejo de cambios en el sistema a través del proceso de desarrollo.

Fases de AUP

La metodología AUP, (al igual que RUP), busca dividir el desarrollo de software en cuatro (4) fases, cada una de las cuales comprende técnicas y disciplinas de desarrollo, que permiten iterar sobre ellas hasta lograr el cumplimiento de los objetivos planteados. Al ser una metodología flexible y ágil, AUP permite que las técnicas que componen cada fase, se

puedan adecuar a las necesidades y realidades del proyecto.

A continuación, se muestra un gráfico con las fases de la metodología AUP (Figura 3.1).

El eje vertical representa la dimensión dinámica, relativa al tiempo y el eje horizontal la dimensión estática.

Implementacion en el Proyecto de Pasantia

AUP es una metodología de cuatro fases, cada una compuesta por un conjunto de técnicas y disciplinas de desarrollo. Por ser una metodología ágil, estas técnicas y disciplinas pueden ser adaptadas a las necesidades, recursos e intereses del proyecto y de la empresa.

Durante el desarrollo de este proyecto se llevaron a cabo las cuatro fases de la metodología: inicio, elaboración, construcción y transición. Las fases de inicio y elaboración se llevaron a cabo en una iteración, la fase de construcción en tres iteraciones por cada plataforma y la fase de transición en una iteración.

Inicio (*Inception*)

Esta es la primera fase del ciclo de vida donde se identifica el alcance inicial del proyecto. El objetivo principal es archivar el consenso de los interesados del proyecto en relación a los objetivos del mismo para obtener financiamiento. [8]

En el contexto de este proyecto los objetivos dispuestos para esta fase están centrados en la definición de los requerimientos del sistema. Las actividades realizadas comprenden la introducción a la empresa, investigación acerca de la situación actual, obtención y modelación inicial de los requerimientos, planteamiento y estudio de las herramientas a utilizar, elaboración de un plan de riesgos, elaboración inicial de un plan de pruebas, elaboración inicial del glosario y la planificación de actividades para las siguientes iteraciones.

Elaboración (*Elaboration*)

La segunda fase consiste en probar la arquitectura potencial del sistema. Se trata de determinar que el equipo pueda desarrollar un sistema que satisfaga los requisitos, por lo que

se realiza una construcción de extremo a extremo del esqueleto del trabajo, conocido como "prototipo de la arquitectura".

Los riesgos se detallan lo suficiente como para entender los riesgos de la arquitectura y para asegurar que exista una comprensión de los alcances de cada requerimiento para que la planificación posterior se puede llevar a cabo. [19]

En este proyecto las actividades comprendidas durante esta fase involucran la identificación de los riesgos técnicos, el modelado de la arquitectura y diseños de prototipos, así como la refinación de documentos desarrollados en la fase anterior.

Construcción (*Construction*)

Esta fase involucra la construcción de un software funcional sobre una base regular e incremental, las cuales cumplan con las prioridades más importantes para los involucrados o usuarios del proyecto. El objetivo de esta fase consiste en desarrollar el sistema hasta el punto que se encuentra listo para la pre-producción de pruebas. [19]

Para este proyecto esta fase presenta tres iteraciones:

- Primera iteración:
  - Diseño de un nuevo protocolo de comunicacion con el servidor.
  - Diseño e implementación del mecanismo de obtención de datos.
- Segunda iteración:
  - Diseño e implementación de almacenamiento en cache.
  - Diseño e implementación del funcionamiento al no tener red de datos.
- Tercera iteración:
  - Diseño e implementación de la lógica de retraso de peticiones.

Transición (*Transition*)

Esta última fase consiste en la validación y despliegue del sistema en su ambiente de producción. Durante esta fase deben hacerse pruebas extensivas, para asegurarse de que el sistema puede ser desplegado de manera segura y eficiente. [19]

En el desarrollo de este proyecto en la fase de transición se realizaron las pruebas necesarias para comprobar el funcionamiento correcto de los componentes para las dos plataformas y la refinación de documentos. El alcance de este proyecto de pasantía sólo abarca hasta la generación de documentos, con la elaboración del informe de pasantía. En este sentido el proceso de puesta en producción queda fuera de las 20 semanas estipuladas.

Capítulo 4

Marco Tecnológico

A continuación, se presentan los aspectos tecnológicos para el desarrollo del proyecto de pasantía, se describen brevemente y se indican su uso como parte de la solución propuesta.

Android

Es un sistema operativo para dispositivos móviles tanto teléfonos celulares como tabletas. Fue desarrollado por Android Inc, firma que fue comprada por Google en el año 2005. Es el más utilizado en el mundo, esto a razón del bajo costo que ofrece a las empresas fabricantes de los dispositivos. Además Android también es personalizable y ligero, cualidades muy importantes y atractivas. La mayoría del código de Android está liberado bajo la licencia Apache.

El sistema operativo Android consta de un núcleo basado en los núcleos de Linux 2.6 y 3.x, con librerías, APIs y middleware escritos en C. El software de aplicación se ejecuta sobre un framework que incluye librerías compatibles con Java basadas en Apache Harmony. Usa Dalvik como máquina virtual, compilando las aplicaciones en tiempo de ejecución.Android usa OpenCore como Framework, SQLite como manejador de base de datos, WebKit como motor de renderizado, OpenGL como interfaz de programación de API gráfica, un administrador de interfaz gráfica, un motor gráfico SGL, una biblioteca estándar de C llamada Bionic y una capa de conexión segura (SSL).[9]

La última versión es la 4.2.1, llamada Jelly Bean (correspondiente al API 17). Salió al mercado en noviembre del año 2012.

Las aplicaciones, tanto pagas como gratuitas, pueden ser descargadas de la tienda virtual Play Store, mediante una aplicación con el mismo nombre.

Java

Es un lenguaje de programación de alto nivel orientado a objetos desarrollado por Sun Microsystems. Las aplicaciones escritas en Java son compiladas a bytecode y pueden ser ejecutadas en cualquier máquina virtual de Java. Gracias a su portabilidad, Java es usado para el desarrollo de aplicaciones móviles para las plataformas Blackberry y Android.

Android ha creado un conjunto de librerías en este lenguaje que, en conjunto

con las librerías propias de Java, permiten el desarrollo de aplicaciones para dispositivos portátiles que tengan instalado el sistema operativo Android. En el portal web http://developer.android.com/reference/packages.html se puede encontrar la referencia de las librerías que soporta una aplicación Android, con la facilidad de filtrarlas por nivel de API.

Además de las librerías propias de Java, y las agregadas por Android, se utilizaron librerías de terceros para poder incluir algunas funcionalidades que no poseen las versiones de los sistemas operativos que el componente debe soportar.

iOS

Es un sistema operativo para dispositivos móviles desarrollado y distribuido por la compañía Apple, para ser usado únicamente en los dispositivos fabricados por la misma firma. Entre estos dispositivos se encuentran el iPhone, iPod Touch, iPad y el Apple TV.

iOS es un derivado de OS X, el cual es un sistema operativo tipo Unix. La estructura de iOS está basado en una abstracción de cuatro capas: la Capa de Núcleo del Sistema Operativo, la Capa de Servicios, la Capa Multimedia y la Capa de Cocoa Touch (framework de interfaz de usuario). Está escrito en C, C++ y Objective-C.

La última versión es la 6.1, disponible desde diciembre del 2012.

Las aplicaciones, tanto pagas como gratuitas, pueden ser descargadas de la tienda virtual App Store, mediante una aplicación con el mismo nombre.

Objective-C

Es un lenguaje de programación orientado a objetos de alto nivel. Su sintaxis es similar a la de Smalltalk. Es usado para el desarrollo de los sistemas operativos de Apple OS X y iOS, así como para las Interfaces de Programación de Aplicaciones (APIs) Cocoa y Cocoa Touch.

Solo se permite el desarrollo de aplicaciones en Objective-C mediante el uso de un IDE llamado Xcode, el cual se puede descargar gratuitamente si se posee el hardware y software de Apple. La última versión de Xcode es la 4.5.2. Xcode permite instalar los simuladores de los dispositivos de las últimas versiones para la realización de las pruebas. La documentación sobre las librerías para el desarrollo de aplicaciones se encuentra en el portal web ttps://developer.apple.com/library/ios/navigation/index.html.

RPC

La Llamada a Procedimiento Remoto (Remote Procedure Call) es un mecanismo para las aplicaciones Cliente-Servidor, desarrollado por Sun Microsystems. En la comunicación participa un procedimiento invocador (cliente) y un procedimiento invocado (servidor). El objetivo de RPC es que el cliente realice una llamada a un procedimiento del servidor como si el mismo sea local.

En lineas generales, para realizar la llamada a un procedimiento remoto, el programa cliente debe enlazarse con un procedimiento de librería llamado *stub* del cliente, que representa el procedimiento del servidor en el espacio del cliente. De la misma forma, el servidor se enlaza con un procedimiento llamado *stub* del servidor. Ambos procedimiento ocultan el hecho de que la llamada no es local.

La comunicación sconsta de los siguientes pasos:

1. El cliente llama a *stub* del cliente. Esta es una llamada local, por lo que los parámetros se colocan en la pila.
2. El *stub* del cliente empaqueta los parámetros en un mensaje y realiza una llamada de sistema para enviar el mensaje.
3. El kernel envía el mensaje desde la máquina cliente a la máquina servidor.
4. El stub del cliente llama al procedimiento servidor con los parámetros.

La respuesta del servidor al cliente se realiza de la misma forma. [11]

XML-RPC

Es una llamada a procedimiento remoto que usa HTTP como protocolo de transporte y XML para la codificación de los mensajes. Permite la transmisión de estructuras de datos complejas, pero la interpretación de las mismas es sencilla. [13]

XML-RPC es la forma de comunicación que tiene actualmente el AdServer MobileX con el componente AdField MobileX. Tanto la petición que hace el cliente como la respuesta que recibe son empaquetadas en un XML, y ambas contienen muchos etiquetas que son inútiles tanto para el cliente como para el servidor.

XML (eXtensible Markup Language)

Es un lenguaje desarrollado por el World Wide Web Consortium (W3C) que es usado para el intercambio de información entre distintas plataformas. Su flexibilidad permite que sea usado para representar cualquier estructura de datos, desde arreglos y árboles hasta, por ejemplo, un servicio web o una base de datos.

JSON-RPC

Es un protocolo de llamada a procedimiento remoto similar a XML-RPC, solo que el formato de codificación es JSON en lugar de XML. Utilizar JSON permite enviar estructuras de datos tan complejas como las que se pueden enviar con XML-RPC, pero el mensaje es más ligero. [14]

Una variación del JSON-RPC es la forma de comunicación propuesta para la comunicación entre el AdServer MobileX y el nuevo componente AdView MobileX. Los argumentos de la solicitud de publicidad no se empaquetarán en un JSON, si no que se enviarán como argumentos al hacer la petición HTTP. La respuesta será un JSON que contenga solo los campos necesarios para mostrar y reportar la publicidad.

JSON (JavaScript Object Notation)

Es un lenguaje que ofrece un formato ligero para el intercambio de información, siendo una alternativa a XML. Su ventaja frente a XML es que su formato permite la creación de un diccionario, por lo que no son necesarias el exceso de banderas que necesita XML para la represantación de las estructuras. En la figura 4.1 se ven los dos formatos representando los mismo datos.

Capítulo 5

Desarrollo

A continuación, se presenta el proceso de desarrollo de la pasantía. Se describen, de manera específica, cada una de las actividades realizadas en cada fase de AUP, las dificultades encontradas y las decisiones tomadas a lo largo del desarrollo.

Face de Inicio

Los objetivos principales de la primera fase son la identificación del alcance inicial del proyecto y la modelación de los requerimientos especificados para el componente. Esta fase fue planificada para ser desarrollada en una iteración.

Actividades propuestas dentro de la fase de inicio

Las actividades consideradas a llevar a cabo en la fase de inicio son las siguientes:

1. Introducción a la empresa.
2. Investigación acerca la situación actual, lo que incluye:
    a) Investigación acerca la publicidad en dispositivos móviles.
    b) Investigación acerca las formas de publicidad entregada por la empresa.
    c) Investigación acerca de componentes de publicidad para móviles en el mercado.
    d) Investigación acerca del AdServer MobileX.
    e) Investigación acerca del AdField MobileX.
3. Modelación inicial de los requerimientos.
4. Planteamiento y estudio de las herramientas a utilizar, así como de los usuarios involucrados.
5. Elaboración inicial de un Plan de Riesgos.
6. Planificación de las actividades para las siguientes iteraciones.

Modelado de Requerimientos de Alto Nivel

El objetivo del modelado de requerimientos es identificar las metas del negocio, desarrollar una visión común y determinar los requerimientos iniciales del componente en un alto nivel.

Para la determinación del alcance inicial del proyecto fue necesaria la construcción de un listado de requerimientos, con la finalidad de obtener una visión acerca de lo qué se pretendía lograr y la identificación de una solución viable. Debido a limitaciones de tiempo y por decisión de los dirigentes de la empresa donde fue desarrollado el proyecto de pasantía, el enfoque del desarrollo de este proyecto estuvo dirigido a requerimientos funcionales.

Luego de establecer los requerimientos, estos se dividieron en tres categorías, dependiendo de la prioridad y las funcionalidades que se desean implementar. Las categorías son: Requerimientos Básicos, Requerimientos Avanzados y Requerimientos Adicionales.

Los Requerimientos Básicos son aquellos que permiten hacer la solicitud de la publicidad y colocarla adecuadamente en los dispositivos portátiles. Además permiten hacer el reporte de las impresiones en pantalla y de los clics hechos soble la publicidad. Los siguientes requerimientos pertenecen a este grupo:

- Definir un nuevo protocolo de comunicación ligero.
- Mostrar de forma efectiva las formas de publicidad manejadas por la empresa.
- Notificar que se realizó la impresión de la creatividad.
- Notificar que se realizó clic en la creatividad y mostrar el contenido.

Los requerimientos Avanzados son aquellos que permiten reutilizar las publicidades descargadas mientras las mismas estén dentro del periodo de validez enviado por el servidor. Estos requerimientos ayudan a resta carga de trabajo al servidor y reducen el tiempo que tarda en aparecer la publicidad. Este estos requerimientos se encuentran:

- Definir mecanismo de almacenamiento.
- Definir mecanismo de rotación.

Los requerimientos Adicionales son aquellos que complementan los requerimientos Básicos y Avanzados para tener un componente que siga las reglas del negocio de la empresa.

- Definir mecanismo de colocación de creatividad si el dispositivo no tiene conexión a red de datos.

- Reducir el trabajo del servidor a la hora de decidir la creatividad a enviar.
- Reporte de la versión del componente y nombre de la aplicación.
- Permitir la segmentación de los usuarios.
- Definir mecanismo de segmentación por país y operadora.

La lista de estos requerimientos se encuentra contemplada en el Documento de Requerimientos del Software especificado en el Apéndice C.Gracias a esta recaudacion se puede tener una idea general acerca las funcionalidades del componente sin enfocarse en los detalles de implementación, y de esta manera tener una mejor idea de qué es lo que se desea implementar. Esta forma de representación permite conocer lo que el usuario necesita sin dejar que las decisiones de implementación interfieran.

Hardware

Para el desarrollo del componente en iOS se utilizó como estación de trabajo una laptop MacBook, con acceso a internet, procesador Intel Core 2 Duo de 2 GHz, 1 GB de memoria RAM, 160 GB de disco duro y sistema operativo Mac OS X 10.6.8 Snow Leopard.

En el caso de la versión de Android, se utilizó una computadora de escritorio de marca Lenovo, con acceso a internet, procesador Intel Premium Dual de 2.40 GHz, 2 GB de memoria RAM, 150 GB de disco duro y sistema operativo de 32 bits Windows XP Professional Version 2002 Service Pack 3.

Software

El lenguaje utilizado para el desarrollo del componente en iOS fue Objective-C, usando como IDE (Entorno de Desarrollo Integrado) Xcode Versión 4.0.

En el caso del componente en Android, se usó como IDE Eclipse 3.7 Indigo, con un plugin de Herramienta de Desarrollo de Android (Android Development Tools), y como lenguaje de programación Java.

Usuarios que participan en el sistema

En el cuadro 5.1 se listan los usuarios que participan en el componente:

Cuadro 5.1: Usuarios que participan en el Sistema

Nombre	Descripción
Administrador	Se encarga de administrar el sistema AdServer MobileX.
Desarrollador de Aplicaciones	Es el creador de la aplicación donde será desplegada la publicidad. Entre sus responsabilidades se encuentra el uso apropiado del componente, para que la publicidad se muestre de la mejor manera y se haga el reporte de manera apropiada.
Desarrollador del la Creatividad	Es el creador de la publicidad desplegada en las aplicaciones.
Desarrollador del Componente	Es el creador del componente para las plataformas móviles utilizadas en la empresa. Entre sus responsabilidades esta crear un componente portable, funcional y extensible que sea fácil de utilizar por el desarrollador de aplicaciones.

Riesgos

Se elaboró una lista de riesgos inicial para la identificación de riesgos y la mitigación temprana de los mismos, evitando retrasos en la terminación de los módulos del sistema. Entre los riesgos listados se encuentran:

- Falla Física sobre la Máquina de Trabajo.
- Falta de Documentación sobre el Protocolo Actual.
- Falta de Documentación sobre el Componente Actual.
- Falta de Tiempo del Administrador del Servidor para Realizar Pruebas.
- Falta de Conocimientos sobre el Software a utilizar
- Falta de Planificación

El plan de riesgos final se encuentra anexado en el Apéndice A.

Plan de Pruebas

Se realizó la versión inicial del plan de pruebas. Este documento incluye una recolección de las pruebas a realizar para la comprobación del correcto funcionamiento del componente.

Fase de Elaboración

El principal objetivo de esta fase es asegurar la factibilidad del componente a desarrollar, a través de un diseño de las funcionalidades a implementar.

Las principales actividades realizadas durante esta fase fueron el diseño de prototipos que cumplieran con los requerimientos iniciales especificados en la fase anterior (a partir de un modelado por lluvia de ideas), configuración del ambiente de desarrollo y la refinación de documentos realizados en la fase de inicio.

Para la realización de esta fase se realizó una investigación exhaustiva acerca el sistema MobileX, tanto del servidor como del cliente, con el objetivo de tener una idea acerca su funcionamiento, y las mejoras necesarias correspondientes al modelo de negocio de la empresa. Esta fase fue planificada para realizarse en una iteración.

Modelado por Lluvia de Ideas

El modelado por lluvia de ideas es un método que comprende la presentación de un conjunto de ideas, que luego son analizadas para concluir la mejor solución de acuerdo a los requerimientos, procediendo a un refinamiento de casos de uso. Esta técnica se llevó a cabo para cada uno de los requerimientos iniciales:

Definir un nuevo protocolo de comunicación ligero

Para la comuniación entre el AdServer MobileX y el componente AdField MobileX se usa XML-RPC. El componente envía un XML con el número de zona y una serie de datos vacios, y recibe un xml, con muchos datos que no utiliza. Como las aplicaciones en los dispositivos móviles se ven afectadas por la cantidad de datos que envían y reciben, el objetivo del nuevo protocolo es que sea lo más simple y reducido que se pueda.

Mantener la comunicación con XML-RPC y reducir los datos enviados

Al evaluar la estructura XML que envía el componente al solicitar la publicidad vimos que actualmente la unica información importante es el número de zona. Los demás datos son vacíos y el servidor no hace uso de ellos. Así que se podría enviar el XML con ese único valor.

Por otra parte, después de evaluar la estructura XML-RPC recibida, donde solo una pequeña parte servía para la impresión y el reporte de las creatividades, definimos una nueva

respuesta que enviaría el servidor. La estructura nueva enviará unicamente los siguientes campos:

1. Dirección url donde se encuentra la creatividad.
2. Dirección url que será llamada cuando se muestre la creatividad en la aplicación.
3. Dirección url que será llamada cuando se haga clic sobre la creatividad en la aplicación.
4. Ancho que mide la creatividad.
5. Alto que mide la creatividad.
6. Identificador identificador del tipo de creatividad.
7. Tipo de contenido.
8. Tipo de campaña.
9. Fecha de vencimiento.
10. Tiempo en cache.

Realizar la comunicacion mediante JSON-RPC y reducir los datos enviados

A pesar de que la cantidad de datos enviados por ambas partes se reduce con la solución propuesta anteriormente, JSON-RPC ofrece una comunicación mucho más ligera, por usar el formato JSON para codificar las estructuras. Por lo tanto esta solución es mucho mejor que la anterior.

Realizar la comunicacion mediante un híbrido del JSON-RPC

El AdServer MobileX es un servidor que realiza diversos servicios, por lo que se busca también quitarle un poco de carga. Enviar unos cuantos argumentos en un JSON generará que el servidor deba parsear esta estructura para obtener los datos que necesita para responder con la creatividad adecuada. Por lo tanto, después de varias discusiones con miembros de la empresa encargados del servidor, se decidió colocar los argumentos en la dirección url a la cual se realiza la peticion de publicidad. Por lo tanto, si se desea enviar un numero de zona, el url lucirá algo como:

http://direccion.del.servidor/procedimientoremoto.php?zoneid=1

Se decidió que la respuesta del servidor fuera un JSON como se planteó anteriormente.

Sin embargo, las plataformas iOS y Android no tienen una librería nativa para la decodificación de archivos JSON en las versiones mínimas que se desean soportar (estas versiones son la 3.2 para Android y la 3.0 para iOS). Por esta razón se deben buscar librerías de terceros que perminan hacer la decodificación.

Reducir el trabajo del servidor a la hora de decidir la creatividad a enviar

Por cada campaña, el servidor mantiene diversos tamaños de una misma creatividad, a fin de enviar al cliente el más apropiado para el dispositivo desde el cual está realizando la petición. Para tomar la decisión de que tamaño enviar, el servidor obtiene de los parámetros provenientes de la solicitud http el agente de usuario (user agent), posteriormente lo busca en una tabla, para obtener los tamaños de la pantalla soportados por el dispositivo. Gracias a los tamaños, busca una creatividad que sea menor al área de la pantalla del dispositivo. Si no encuentra el agente de usuario en la lista, envia una creatividad por defecto, la cual puede no ser la más apropiada para el dispositivo.

La tabla con la lista de los agentes de usuario es llenada a mano por el administrador del servidor. Esta lista debe ser actualizada cada vez que sale un nuevo sistema operativo o un navegador (o una nueva versión de los que ya existen).

La solución a este problema es que se envíe en los argumentos de la petición de publicidad el alto y el ancho de la pantalla del dispositvo, de manera de quitarle al servidor el paso de buscar en la lista. Por lo tanto, el url lucirá como:

http://direccion.del.servidor/procedimientoremoto.php?zoneid=1&width=1&height=1

Reporte de la versión del componente y nombre de la aplicación

Para los datos estadísticos que maneja el servidor con respecto a la utilización del componente por parte de las diferentes plataformas y aplicaciones, se desea que el servidor reciba por parte del cliente el número de la versión del componente y el nombre de la aplicación que lo está usando. Después de discutir con otros miembros de la empresa especializados en el servidor, se decidió que la mejor manera de enviar estos datos es agregarlos al agente de usuario que se envía al servidor. Por lo tanto, un agente de usuario lucirá de la sigueinte manera:

<Agente_de_Usuario> **MobileX/<Número_Versión> <Nombre_Aplicación>**

Permitir la segmentación de los usuarios

El sistema AdServer MobileX permite el registro de usuarios a través de las aplicaciones para disfrutar de algunos servicios que ofrece. Al hacer este registro el servidor envía al cliente una cadena de caracteres que sirve para realizar la segmentación de los usuarios. Se desea que si se posee esta cookie, se pueda enviar al servidor, de manera de que pueda enviar inteligentemente la publicidad, es decir, si en una zona se tiene una publicidad de una zapatería para niños y otra de una tienda de deportes, se envíe la primera si la persona es una mujer entre los 30 y los 40 años, mientras que es mejor enviar la segunda si la persona es un hombre entre los 18 y los 60 años.

Se decidió enviar este valor en la solicitud que se hace al servidor como un argumento más. Por lo tanto, el url lucirá como:

http://direccion.del.servidor/procedimientoremoto.php?zoneid=1&cookie=C00K13

Definir mecanismo de segmentación por país y operadora

Mobile Media Netowrks ofrece sus servicios a diferentes países de America Latina, por lo que es necesario lograr segmentar la entrega de las campañas por países, o por operadoras. Por ejemplo, una empresa de telefonía presente en varios países podría colocar una creatividad anunciando los nuevos precios de un determinado plan de datos. Como la moneda varía según el país, se desea que aparezca la creatividad con la moneda apropiada según la ubicación. Además, es probable que la empresa desee que esta campaña solo aparezca en dispositivos que pertenecen a esta misma operadora.

Una de las soluciones planteadas fue utilizar las cookies recibidas al realizar el registro, puesto que uno de los datos de la cookie es el país, sin embargo, esto no es viable porque muchas de las aplicaciones de móviles no tienen un mecanismo de registro.

Otra de las posibles soluciones es utilizar las librerías nativas de los dispositivos para obtener el MCC y el MNC. El Código de País del Dispositivo (Mobile Country Code, o MCC) es un número único que se asigna a cada país. Por ejemplo, si el usuario está en Venezuela, su MCC es 734. Por otra parte, el Código de Red del Dispositivo (Mobile Network Code, o MNC) es un número único que se asigna a una operadora en un determinado país. Sin el MCC, no es posible saber cual operadora es la que se está consultando. Envíando el MCC y el MNC como argumentos tanto en la petición de publicidad como en el reporte de la impresión y el clic, el servidor puede generar estadísticas y reportes interesantes.

Sin embargo, en varias discusiones con otros miembros de la empresa relacionados a la

administración del servidor y la colocación de la publicidad, se decidió también enviar el número ip del dispositivo. Esta decisión está derivada de que los portales web no pueden obtener el mcc, ni el mnc, por lo que se necesita que ellos también puedan enviar datos sobre su ubicación. La ventaja de usar el ip es que todos los dispositivos y portales web pueden enviar este dato, la desventaja es consultar el país a partir de un ip significa consulta una tabla de millones de filas. Por lo tanto, el servidor usará primero el MCC y el MNC si estos valores no son vacíos.

De esta manera, el url de solicitud lucirá como:

http://direccion.del.servidor/procedimientoremoto.php?zoneid=1mcc=734&mnc=4&ip=192.168.0.0

Mostrar de forma efectiva las formas de publicidad manejadas por la empresa

Mobile Media Networks maneja actualmente solo dos formatos de creatividades, estos son jpg y gif. En cualquier plataforma se pueden mostrar las imagenes planas sin ningún inconveniente (este es el caso de las imagenes jpg). Sin embargo, las plataformas iOS y Android no tienen una librería nativa para mostrar contenido animado (contenido gif) en las versiones mínimas que se desean soportar.Por lo tanto, se deben buscar librerías de terceros que permitan realizar la animación de las imagenes.

Notificar que se realizó la impresión de la creatividad

En el componente AdField, el reporte de la impresión de la creatividad se realiza haciendo una petición HTTP con la dirección url que envía el AdServer. Esta dirección es generada por el servidor, por lo que contiene los argumentos necesarios para hacer el reporte correcto. Si se le agregan argumentos extra, el servidor los ignora. Crear una nueva dirección a mano sería ineficiente.Por lo tanto, se usará la misma forma de reporte que el AdField.

Notificar que se realizó clic en la creatividad y mostrar el contenido

Al igual que la dirección de reporte, la dirección de clic es enviada por el servidor conteniendo todos los argumentos necesarios para mostrar el contenido del portal web asociado a la creatividad, así como la realización del reporte. Cuando se hace clic a la publicidad en una aplicación, la misma debería abrir un navegador web con la información detallada de lo que se quiere mostrar. Por ejemplo, si la creatividad es de un banco, al darle

clic posiblemente lleve al portal web del mismo banco. El reporte se hace de la misma forma que en el caso de una impresión, es decir, mediante una petición HTTP.

En las investigaciones hechas de las dos plataformas, se descubrió que ambas tienen una librería para poder mostrar en un navegador el contenido de una dirección url. Internamente la librería debe hacer una petición HTTP, así que no se requiere de ningún mecanismo extra para el reporte.

Definir mecanismo de almacenamiento

Tener un mecanismo de almacenamiento de las creatividades y sus datos asociados permite que la obtención de la publicidad sea más rápida, ya que se estaría sacando del mismo dispositivo, en lugar de depender de la conexión de datos y su velocidad. Ambas plataformas permiten almacenar elementos en el directorio propio de la aplicación, en un directorio cache, o mediante otros mecanismos propios como bases de datos o listas de preferencias.

Viendo las ventajas y desventajas de cada uno se decidió que las creatividades se guarden en el directorio cache. La ventaja es la rapidez de carga que se tendrá de la creatividad. La desventaja es que el directorio cache de la aplicación podría ser borrado por decisión del sistema operativo; sin embargo, esto no sería un problema, ya que la creatividad no es información sensible para la aplicación.

Con respecto al almacenamiento de las características de las creatividades almacenadas, se decidió tener una lista de preferencias. Ambas plataformas ofrecen librerías para el acceso y escritura de los datos, donde el usuario no tiene que manejar la estructura verdadera. Además, son apropiadas para mantener pocos datos almacenados.

Definir mecanismo de rotación

Debido a que las creatividades y su información relacionada se desean almacenar en el dispositivo para quitarle carga al servidor y reducir el tiempo de espera por la publicidad, se necesita de un mecanismo que establezca cuando se debe pedir una nueva creatividad. El servidor enviará la fecha de expiración de la creatividad, cada vez que se va a mostrar, se chequea esta fecha. Si venció, la creatividad se descarta y se pide otra. En caso contrario se considera a la creatividad como vigente y se mostrará.

Además del tiempo de expiración, el servidor también envía el tiempo en cache, el cual determina el tiempo que debe usarse una creatividad antes de pedir otra nueva al servidor. Esto se hace porque la mayoría de las campañas duran más de una semana sin cambiar de creatividad, por lo que muchas veces se piden creatividades que ya se tienen almacenadas.

Este tiempo le indica al cliente que no haga otra solicitud hasta que pase el tiempo enviado.

Cuando se pida una creatividad de cache, se revisará primero si es vigente, es decir, si no ha expirado. Si es vigente, se revisará si es válida. Ser válida significa que, desde el momento que se descargó, no se ha pasado el tiempo que envío el cache de espera (tiempo de cache). Por lo tanto, si la creatividad es válida, se mostrará, pero no se pedirán nuevas creatividades al servidor. De no ser válida (pero si vigente) se mostrará la creatividad pero se pedirá otra al servidor. Esta nueva creatividad se almacenará en cache como todas las anteriores, y se colocará al final de una cola creada con las listas de preferencias. La próxima vez que se ingrese a la aplicación, la creatividad que debe aparecer es la última descargada.

Definir mecanismo de colocación de creatividad si el dispositivo no tiene conexión a red de datos

Uno de los grandes problemas cuando los dispositivos no poseen red de datos es que en la aplicación queda un espacio inutilizado porque la creatividad no se muestra. Se puede aprovechar el hecho de tener un mecanismo de almancenamiento para evitar que esto suceda.

Refinamiento del Plan de Pruebas

En esta fase se realizó un refinamiento al plan de pruebas adaptado a las estructuras definidas en base a las secciones anteriores. El plan de pruebas puede ser visualizado en el apéndice F.

Riesgos Técnicos

Al finalizar esta fase se determinaron los riesgos técnicos relacionados con el diseño proveniente del modelado por lluvia de ideas:

1. Soporte del componente de nuevas formas publicitarias. Por ejemplo: Rich Media.
2. Soporte del componente de nuevas necesidades del mercado.

Fase de Construcción

El objetivo de la fase de construcción es desarrollar los componentes publicitarios hasta el punto que se encuentre listo para la pre-producción de pruebas. El énfasis en las fases

anteriores fue priorizar y comprender los requerimientos, utilizando el modelado para atacar una solución. En esta fase se recurre a la codificación, presentando primeras versiones del componente que muestra la publicidad en las aplicaciones.

Esta fase fue realizada en tres iteraciones por cada plataforma desarrollada.

A continuación se hace una descripción de las decisiones tomadas y los procedimientos establecidos, haciendo una distinción en el caso que amerite por plataforma, ya que en general el componente se comporta de igual manera en ambas plataformas..

Primera Iteración

Diseño e implementación de un mecanismo para mostrar las creatividades

Antes de específicar como es el proceso de muestra de creatividades propio por plataforma, se debe mencionar que las interfaces de usuario no pueden tener componentes que se bloqueen por más de cierto tiempo, porque entorpecen el funcionamiento de la aplicación o causan que la aplicación se cierre forzosamente. Por lo tanto es necesario que desde el comienzo desde la primera llamada al componente, se cree un nuevo hilo de ejecución, que permita realizar toda la lógica del componente sin causar ningún efecto desfavorable a la aplicación principal.

iOS

Una creatividad podría catalogar en iOS como una Vista de Imagen (Image View) sin embargo, el componente tendrá muchas más funcionalidades que una imagen normal. Investigando arduamente las formas de agregar componentes personalizados a la interfaz de usuario, se descubrió que la manera más sencilla es la creación de un delegado (Delegate) y controlador (Controller).

El programa delegado debe ser una extensión de una Vista Imagen, para luego poder asociar la Vista Imagen que se coloque en la interfaz de usuario a la nueva que se está definiendo. El delegado se encargará de realizar los siguientes pasos:

1. Agregar a la dirección url donde se solicita la publicidad los argumentos obligatorios y opcionales. El único argumento obligatorio es el número de zona. Los demás argumentos definidos en la lista de requerimientos serán opcionales.
2. Establecer una conexión HTTP con el servidor.
3. Obtener la respuesta del servidor, la cual está en formato JSON y aplicarle un decodificador.

4. Tomar los campos obtenidos con el decodificador y enviarlo al controlador.

Es importante resaltar que la versión de iOS que se quiere soportar no tiene un codificador/decodificador nativo de JSON, por lo que se usó la librería de terceros JSONKit. JSONKit está licenciado bajo BSD y Apache 2.0.[15]

El programa controlador es el que se encargará de modificar la interfaz de usuario. A partir de los argumentos recibidos desde el programa delegado, deberá descargar la creatividad y colocarla de acuerdo a las dimensiones originales. Dado que las dos formas de creatividad que maneja actualmente la compañía son las imagenes en formato jpg y gif, el controlador debe colocar cualquiera de las dos sin problemas. La versión de iOS que se quiere soportar tampoco tiene una librería nativa para el manejo de imágenes tipo gif, por lo que se necesitó de una libreria de terceros, llamada Animated Gif [16], para poder colocarlos. Esta librería coloca cuadro por cuadro del gif, sin dejar el primero cuadro de base, como hacen otras librerías.

Android

Como se estableció anteriormente, el componente publicitario tiene una lógica oculta al usuario donde se implementan más funcionalidades que las obvias. Por lo tanto, aunque una creatividad podría catalogar como una Vista Imagen (Image View) en Android. Luego de realizar una investigación exhaustiva sobre las formas de agregar componentes personalizados a la interfaz de usuario, se descubrió que, en el caso de Android, la manera más simple es la creación de un componente de interfaz y un programa controlador.

Un componente de interfaz en Android es codigo XML que establece las características de un componente. Cuando se tienen componentes personalizados, queda a cargo del programa controlador definir cual será el comportamiento del mismo. Para establecer cual será el controlador del componente personalizado, el nombre del mismo debe ser la ruta completa del nombre del programa que será el controlador.

El programa controlador debe ser una extensión de una Vista, para poder realizar las modificaciones al componente personalizado. El programa controlador realizará las mismas ladores que el programa delegado de iOS definido en esta misma sección.

Como el programa controlador se encargará de modificar la interfaz de usuario, debe ser capaz de mostrar en el mismo espacio cualquiera de las formas de publicidad manejadas por la empresa. Para la versión de Android que se quiere soportar no existe una librería nativa para el manejo de imágenes tipo gif, por lo que se necesitó de una libreria de terceros, llamada GifView [17]. Esta libreria coloca el primer cuadro fijo y coloca uno por uno los demás sobre este.

Diseño e implementación de un mecanismo para el reporte de la impresión

Tanto para iOS como para Android, el programa controlador debe ser el encargado de llamar a una función que realice una petición HTTP al servidor cuando la creatividad sea descargada y mostrada en la interfaz de usuario. Como la dirección url ya trae los argumentos que necesita el servidor para hacer el reporte correcto, solo se debe hacer la llamada.

Diseño e implementación de un mecanismo para el reporte del clic y la colocación apropiada del contenido

iOS

Con respecto al reporte del clic, este procedimiento es más delicado, no solo sirve realizar una petición HTTP, puesto que la dirección de clic trae como argumento la dirección del portal web al cual se redireccionará el navegador.

Leyendo la documentación, no existe para la versión de iOS que se quiere soportar un componente que permita abrir un explorador dentro de la aplicación. Sin embargo, si se puede cargar la información en una Vista Web sencilla dentro de la aplicación o lanzar el explorador predeterminado. La ventaja de la primera opción es que no se abandona la aplicación al hacer clic sobre la creatividad, pero la Vista Web no provee las interfaz de un navegador. La ventaja del explorador es que se tienen todas las funciones de navegador que permiten al usuario disfrutar de una buena interfaz, pero el navegador se abre fuera de la aplicación. Se decidió utilizar la segunda opción.

Al lanzar un navegador con una dirección url, el mismo realizará la llamada HTTP, por lo que se estará reportando el clic automáticamente.

Android

Al igual que en iOS, no existe un componente en Android que permita abrir un navegador dentro de la aplicación. Sin embargo, por la estructura de Android, no es necesario. Android maneja cada interfaz por medio de Actividades, cuando se quiere lanzar un navegador web, se crea una nueva actividad, que se empila en la lista de actividades. Si la persona decide salir del navegador, entonces este se quitará de la pila y la persona regresará a la pantalla donde estaba la aplicación antes de hacer clic en el banner.

A pesar de esta ventaja, se investigaron otras formas de colocar contenido web dentro de la aplicación. Al igual que iOS, Android tiene una Vista Web, la cual permite cargar

de manera sencilla contenido web. Sin embargo, no se disfruta de la interfaz propia de un navegador y la implementación es muy básica así que el contenido no se ve atractivo.

Por lo tanto, se decidió utilizar la primera opción, lanzar un navegador. Al hacer esto, se realizará la llamada HTTP, y por lo tanto, se reportará el clic automáticamente.

Segunda Iteración

Diseño e implementación de un mecanismo de almacenamiento

El objetivo que se busca lograr al almacenar la publicidad en el dispositivo es reducir la cantidad de peticiones que se hacen al servidor. Además, se busca que el tiempo que transcurre entre que la aplicación se abre y la creatividad se muestre sea el mínimo.

Almacenar la publicidad implica no solo descargar la creatividad y almacenarla, sino también todos los datos asociados a ella. Por lo tanto se debe definir un mecanismo para el almacenamiento de la imagen y otro para los datos.

iOS

Para el caso del almacenamiento de las creatividades, se usará el directorio Cache de la aplicación, lo que permite un acceso rápido a los datos. El nombre del archivo corresponderá con identificador de la creatividad (bannerID) para poder acceder a él posteriormente.

Con respecto a las características, necesitamos un mecanismo que permita almacenar arreglos, diccionarios y tipos de datos básicos. Las opciones que proporciona iOS para la persistencia de datos son:

1. *NSUserDefaults*: Están basados en las listas de preferencias. Permite almacenar información sencilla, como las prefencias de los usuarios de una aplicación.
2. *Plist*: Permite almacenar datos en una lista de preferencias. El acceso es más rápido que en las Preferencias de Usuarios (NSUserDefaults).
3. *SQLite*: Es un archivo loca que actía como servidor de base de datos. No es útil si se quieren hacer muchos accesos concurrentes.
4. *Core Data*: Permite almacenar datos un poco más complejos. Usa SQLite como manejador de base de datos por defecto.

Los datos de la creatividad no son sensibles, es decir, la aplicación seguirá ejecutandose normalmente si se borran y no tienen información relevante sobre el usuario. De las cuatro

opciones anteriores, la más cercana a las necesidades en la opción dos, utilizar una plist. Las listas de preferencias de iOS permiten almacenar tipos básicos de datos, así como arreglos y diccionarios.

Android

Al igual que iOS, Android permite el almacenamiento de archivos en el directorio Cache de la aplicación, permitiendo un acceso rápido a los datos. De igual manera se usará el identificador de la creatividad (bannerID) como nombre del archivo.

Las opciones del almacenamiento que ofrece Android son las siguientes:

1. *Shared Preferences*: Permite almacenar datos privados de los tipos primitivos de la forma clave-valor.
2. *Internal Storage*: Permite almacenar datos privados en la memoria del dispositivo.
3. *External Storage*: Permite almacenar datos en memorias externas.
4. *SQLite Databases*: Permite almacenar datos en una base de datos privada.
5. *Network Connection*: Permite almacenar datos en un servidor propio.

La opcion número uno es la que más se acerca a las necesidades del almacenamiento. El único problema es que solo permite almacenar los tipos de datos primitivos, es decir, no se pueden crear diccionarios ni arreglos, por lo que se debe disponer de más de un archivo de preferencias.

Por lo tanto, para una zona existirá un archivo principal, donde se guardarán los números identificadores de las creatividades descargadas, y un archivo adicional con las características y datos importantes de la creatividad en específico.

Diseño e implementación de un mecanismo de rotación

Tener un mecanismo de rotación de publicidad permite, en conjunto con el mecanismo de almacenamiento, reducir la cantidad de peticiones al servidor y reducir los tiempos de carga. No se debe confundir el termino rotación con el hecho de cambiar las creatividades en una pantalla cada cierto tiempo. Esto no fue definido como requerimiento, por lo tanto, no fue implementado.

La rotación de las creatividades debe hacerse en los archivos donde se mantiene la lógica de la persistencia. Es en estos archivos, mencionados en el punto anterior, donde están los datos que permiten saber cuales son las creatividades válidas.

Al momento de descargar una nueva publicidad, esta se debe colocar al final de la lista de creatividades, y si se llegó al máximo, borrar la primera de la lista y correr todas una posición. A pesar de que suena a que hay muchas creatividades almacenadas, realmente una zona tendrá máximo 2, la que se mostrará al abrir la aplicación y la que se descargará para ser mostrada la siguiente vez que se abra la aplicación. Además de mantener las listas en orden, uno de los datos que se deben almacenar de la creatividad y que es recibido por el servidor es el tiempo de descarga. Este tiempo, en conjunto con el Tiempo de Cache y la Fecha de Vencimiento que envía el servidor, permite saber si una creatividad es válida, vigente o si venció, como se explicó en la sección 5.2.1.10.

En el momento en que una creatividad esté vencida, o sea eliminada de la lista, se borrarán todos los datos y archivos relacionados a ella. Si la creatividad que se descarga nueva ya está en la lista, solo se colocará al final, pero no se descargarán las imágenes de nuevo.

Tercera Iteración

Diseño e implementación de un mecanismo para el comportamiento cuando no se tiene red de datos

Existe la posibilidad de que un dispositivo móvil no tenga acceso a una red de datos en un momento en específico, pero esto no necesariamente causa que el usuario no pueda usar una aplicación. Muchas aplicaciones pueden ser usadas sin necesidad de tener internet. Sin embargo, cuando se diseña una aplicacion, se asigna un espacio de la interfaz de usuario para mostrar el contenido de la publicidad, y sin red de datos este espacio queda inutilizado.

Aprovechando el mecanismo de almacenamiento, se puede disponer de publicidad aunque no se tenga red de datos. Mientras que la creatividad que está almacenada no venza, podrá mostrarse, pero la funcionalidad del clic no servirá puesto que no hay internet.

Se consideró el caso de querer conocer la cantidad de impresiones que se hacen aunque el dispositivo este sin red, por lo que se almacenará una nueva característica a la información almacenada. Una vez que se tenga acceso a la red, se enviará al servidor este dato. Esta característica es opcional, ya que hay aplicaciones que no funcionan sin red de datos y la publicidad que se mostró no es de relevancia en los reportes.

iOS

A pesar de que Android tiene una librería para saber si en un determinado momento el dispositivo tiene conexión, iOS no cuenta con una para las versiones que se necesitan soportar. Para poder conocer el estado de la red, se necesitó utilizar un componente que no se encuentra en las librerías de iOS llamada Reachability, la cual es distribuida por Apple Inc.

Diseño e implementación de un mecanismo para la segmentación

El servidor ofrece como uno de sus servicios la posibilidad de que los usuarios se regristren. Al realizar el registo, el servidor retorna una cadena de caracteres o galleta (cookie), que contiene los datos de registro de una manera sencilla y corta.

El componente debe permitir agregar esta galleta a la petición que se le hará al servidor. Si el servidor recibe la galleta, podrá enviar una publicidad más apropiada al usuario. La solución propuesta es usar la lista de preferencias (plist) para almacenar este valor en el caso de iOS, y una lista de preferencias compartidas para Android. La galleta se almacenará bajo la etiqueta *cookie,* permitiendo que el componente envíe la galleta que tenga almacenada en cada solicitud.

Si no se tiene almacenada una galleta, el argumento en la solicitud será vacío, y por lo tanto el servidor lo ignorará. Si los registros cambian con el tiempo y el formato de la galleta también, solo se tendrá que sustituir el valor anterior por el nuevo.

Otro servicio que ofrece Mobile Media es enviar publicidad especifica unicamente a los dispositivos o portales que se encuentren en un país o que pertenezcan a una operadora. Para poder hacer esto, el componente debe enviar algún dato que permita conocer estos valores. Como se discutió en la sección 5.2.1.5, estos valores son el Código de País del Dispositivo (MCC), el Código de la Operadora del Dispositivo (MNC) y la dirección IP. Gracias a las librerías propias de las plataformas, estos datos se pueden obtener fácilmente.

Por último, también es necesario conocer el dispositivo que realiza la solicitud, ya que el servidor ofrece el servicio de enviar una publicidad a un dispositivo en específico. Gracias al agente de usuario, podemos conocer cual es la marca y modelo del dispositivo. En un futuro, el agente de usuario también servirá para conocer si un dispositivo soporta formas de publicidad nuevas.

Además del agente de usuario se busca incluir también la versión del componente y el nombre de la aplicación, lo que generará estadísticas más robustas y la ventaja de enviar publicidad dirigida específicamente a los usuarios de cierta aplicación. Este mecanismo fue

muy fácil de implementar, ya que tanto iOS como Android ofrecen librerías que permiten obtener estos valores rápidamente.

Fase de Transición

En esta última fase se realizaron las pruebas de revisión de ambos componentes a partir del plan de pruebas creado, dando como resultado un prototipo 100 % funcional para el componente AdView MobileX de Android y un prototipo 100 % funcional para el componente AdViewMobileX de iOS.

Por otra parte, se finalizó la documentación general de los componentes y el refinamiento de los documentos. También se realizaron los documentos de utilización del componente, los cuales se encuentran en los apéndices y .Así mismo se realizaron las presentaciones finales del componente a los involucrados con el mismo, entregando los artefactos que lo comprenden y toda la documentación descrita en este informe.

Conclusiones y Recomendaciones

La empresa Mobile Media Networks utiliza el sistema AdServer MobileX como una plataforma publicitaria para la distribución de publicidad en portales web y dispositvos móviles. Para la colocación simple y apropiada en las aplicaciones, el AdServer MobileX se comunica con un componente publicitario llamada AdView MobileX (o AdField, para la versión de Blackberry). Este componente hace transparente al usuario toda la lógica de negocios que está implementada, ofreciendo al AdServer datos importantes para la segmentación y el reporte, además de reducir la carga del mismo.

El presente proyecto consistió en el diseño de un protocolo de comunicación simple y compacto, y el diseño y la implementación de un componente que cumple con los requerimientos establecidos al comienzo de la pasantía. Estos requerimientos se dividieron en tres categorías: Requerimientos Básicos, Requerimientos Avanzados y Requerimientos Adicionales.

A partir de los requerimientos básicos se implementó un componente que tiene la capacidad de construir una petición a un servidor, realizar la comunicación e interpretar la respuesta. A partir de esta respuesta coloca la creatividad en el lugar y de la manera adecuada, soportando todas las formas publicitarias que maneja la empresa y sin afectar de ninguna manera la interfaz de usuario.

Los requerimientos avanzados aportaron al componente un poco más de inteligencia, siendo capaz de decidir cuando pedir una publicidad y cuando aprovechar lo máximo una que ya se descargó. Además, las creatividades se cargan inmediatamente al momento de abrir la aplicación, haciendo que el componente no deje espacios inútiles en la interfaz.

Los últimos requerimientos aportaron al componente la posibilidad de contribuir con información para las estadísticas y reportes que genera el servidor, además de aportar la lógica necesaria para mostrar la publicidad a pesar de no tener conexión de red.

La puesta en producción de los módulos del servidor que permitan realizar la comunicación con el nuevo protocolo, o que permitan obtener nueva información que envía el componente no están en el alcance del proyecto. Esto obedece a una decisión que debe tomar la gerencia de la empresa en conjunto con los miembros de cargos más altos.

La metodología AUP contribuyó con la planificación del proyecto y a la presentación de la información, la cual servirá a futuros desarrolladores que se involucren con los dos componentes implementados.

La experiencia obtenida durante el desarrollo de este proyecto de pasantía comprende nuevos elementos de aprendizaje vinculados tanto a las nuevas tecnologías utilizadas como al desenvolvimiento laboral y las nuevas relaciones construidas en la empresa.

Para terminar, se presentan algunas recomendaciones:

- Incorporación de funcionalidades para poder colocar elementos de contenido interactivo Rich Media.
- Incorporación de una funcionalidad que consista en que cuando el usuario haga clic en una creatividad que promociona un servicio de otra aplicación, chequee si la apliación existe, y de existir, lance la aplicación en la sección donde se encuentra la venta del servicio.
- Realizar la documentación de los nuevos protocolos y componentes, pero tanbién de los anteriores y de las nuevas funcionalidades que se incorporarán al servidor. El sistema MobileX en conjunto es muy complejo, con clientes publicitarios escritos para distintas plataformas y un servidor muy amplio con muchos servicios ofrecidos.

Referencias Bibliográficas

[1] Hitos y trayectoria: Qué hemos conseguido hasta hoy. Disponible en http://www.adsmediamobile.com/es/compania/hitos-y-trayectoria, consultado el 19 de noviembre de 2012. 1.1.1

[2] AdsMedia Mobile Advertising. "Misión y visión". Presentación interna. 1.1.3

[3] Mobile Media Networks. "Pasante de Tecnología". Presentación interna. 1.1.4.1

[4] Client Server Networks. http://compnetworking.about.com/od/basicnetworkingfaqs/a/client-server.htm, consultado el 11 de agosto de 2012. 2.2.1

[5] Agente de Ususario. http://whatsmyuseragent.com/, consultado el 25 de diciembre de 2012. (document), 2.1

[6] Metodología AUP (Agile Unified Process – Proceso Unificado Ágil) para Implementación del ERP ADempiere – Parte 1. http://ubuntu-adempiere.blogspot.com/2011/09/metodologia-aup-agile-unified-process.html 3.1

[7] The Agile Unified Process (AUP), http://www.ambysoft.com/unifiedprocess/agileUP.html, consultado el 16 de agosto de 2012. 3.1

[8] Fases del AUP. http://cgi.una.ac.cr/AUP/html/phases.html, consultado el 10 de agosto de 2012. 3.3.1

[9] Android. http://www.android.com/about/, consultado el 21 de noviembre de 2012. 4.1

[10] JSON http://www.json.org/json-es.html, consultado el 21 de noviembre de 2012.

[11] A.Tanenbaum. Redes de Computadoreas. 4ta edición. Prentice Hall, 2003. 4.3

[12] W.Stallings. Sistemas Operativos. Aspectos Internos y Principios de Diseño. 5ta edición. Pearson 2005.

[13] http://xmlrpc.scripting.com/, consultado el 21 de noviembre de 2012. 4.3.1

[14] http://json-rpc.org/, consultado el 21 de noviembre de 2012. 4.3.2

[15] JSONKit. https://github.com/johnezang/JSONKit, consultado el 27 de diciembre de 2012. 5.3.1.1

[16] AnimatedGif. https://github.com/scspijker/iOS_AnimatedGif, consultado el 27 de diciembre de 2012. 5.3.1.1

[17] GifView. http://code.google.com/p/android-gifview/, consultado el 27 de diciembre de 2012. 5.3.1.1

[18] Data Storage. http://developer.android.com/guide/topics/data/data-storage.html, consultado el 29 de diciembre de 2012.

[19] Data Persistence. http://mobile.tutsplus.com/tutorials/iphone/iphone-sdk_store-data/, consultado el 29 de diciembre de 2012.

Apéndice A

Documento de Lista de Riesgos!

Apéndice B

Documento Plan de Pruebas

Apéndice C

Documento Requerimiento del Software

Apéndice D

Documento Utilización del Componente para iOS

Apéndice E

Documento Utilización del Componente para Android