Geolocalización de Sitios de Interés Para Aplicaciones Móviles

G-SIAM

Carpeta Principal

Montevideo, Mayo 2012

A/S Antonio Porcelli
A/S Martín Loureiro
Tutor: Ing. Martín Cabrera

VERSIÓN 1.0

Universidad de la Empresa
Soriano 959 - Tel.: 2900 2442
Montevideo - Uruguay | info@ude.edu.uy

Resumen

El proyecto Gsiam surge por la inquietud del equipo en introducirnos en el mercado de los dispositivos móviles el cual está teniendo un crecimiento exponencial y cada vez más usuarios acceden a Internet por medio de sus teléfonos celulares [*ref.001*].
El proyecto se enfoco en el sistema operativo Android el cual empieza a perfilarse como líder de mercado.
En la primera etapa del proyecto el equipo se enfoco en estudiar la tecnología, nueva para los integrantes y en realizar un documento que refleje dicha investigación. Luego se realizo una prueba de concepto con una funcionalidad básica del sistema, obtener los sitios dada una posición. El poder hacer esta prueba fue de vital importancia ya que nos abrió el camino para el desarrollo futuro.
Luego se desarrollo la aplicación empleando un modelo iterativo e incremental el cual nos facilito la definición de requerimientos y el temprano hallazgo de errores. El producto final de esta etapa fue un prototipo para dispositivos móviles con sistema operativo Android que soporta las versiones 2.2 y 2.3 que permite al usuario saber que sitios de interés tiene a su alrededor dándole la posibilidad de comentar, puntuar e incluso publicar sus comentarios en Facebook.

Agradecimientos

Queremos agradecer a todas las personas que directa o indirectamente han colaborado para llevar adelante este proyecto. Familiares y amigos han sido pilares fundamentales que han brindado un grado invalorable de compresión y apoyo; sobre todo la familia cuyo esfuerzo y trabajo diario han sido imprescindibles para conseguir todos nuestros retos.

Nos gustaría hacer una mención especial a Martín Cabrera por toda la ayuda recibida en cada uno de los objetivos propuestos así como también su buena disposición para las reuniones en horarios complicados y en fin de semana.

Índice de Capítulos

Índice de Contenidos

Capítulo I. Introducción

El presente documento describe los aspectos principales del proyecto Gsiam. Se trata de un proyecto de grado de la carrera de Licenciatura en Informática desarrollado entre Mayo del 2011 y Febrero del 2012. El equipo está integrado por A/S Antonio Porcelli, A/S Martín Loureiro con la tutoría del Ing. Martín Cabrera.
El proyecto Gsiam consiste en la Geolocalización de sitios de interés a un radio dado de nuestra ubicación. Mediante el GPS [*ref.002*] de los dispositivos móviles o la conexión a Internet, ya sea por medio de wifi o red de datos, estos dispositivos son capaces de determinar la ubicación del usuario siendo este nuestro input más importante.
A demás de poder saber que sitios tengo a mis alrededores puedo saber que comentarios hay sobre ese sitio. Esto lo creemos de importancia ya que nos da cierta información de ese lugar permitiéndonos elegir de una forma más acertada.
También podemos hacer amigos y armar nuestra propia red recomendando sitios y compartiendo información.
El proyecto está organizado en una carpeta principal la cual contiene los documentos fundamentales que se realizaron a lo largo de la tesis, tales como documentos de ingeniería de software, arquitectura y especificación de requerimientos entre otros. En la carpeta de anexos irán todos los documentos restantes que no fueron incluidos en la carpeta principal pero que fueron generados a lo largo del proyecto y es de importancia su presencia.

1 Como leer el documento

En el índice se puede ver cómo está organizado el documento para realizar una lectura adecuada. Para esto se dispone de un índice de capítulos con los títulos generales y un índice detallado con la especificación de cada uno.
Capítulo I Introducción: Presenta una introducción al proyecto y una explicación de cómo leer el presente documento.
Capítulo II Gestión de Proyecto: Este capítulo detalla todo lo relacionado a la ingeniería de software aplicada al proyecto. Se documentan las decisiones tomadas basándose en principios de ingeniería de software aprendidos durante la carrera.
Capítulo III Investigación: Se presenta toda la investigación realizada durante el proyecto. Se detalla los frameworks utilizados y el porqué de la elección de estos. Más información de este capítulo se podrá encontrar en los anexos.
Capítulo IV Implementación: Este capítulo presenta todo lo referente a los requerimientos desarrollados y el documento de arquitectura desarrollado.

Capítulo V Trabajos Futuros: Se presentan los posibles desarrollos futuros a implementar y los requerimientos que no pudieron ser incluidos en el proyecto por razones de tiempo.
Capítulo VI Resultados y Conclusiones: Conclusiones generales de la realización del proyecto.
Capítulo VII Apéndice: Bibliografía, glosario y referencias.

Capítulo II. Gestión de Proyecto

En esta sección se expone todo lo relacionado a la gestión del proyecto, cual fue el modelo elegido y porque, análisis de riesgo, plan de pruebas, plan de SQA, plan de GCF (gestión de configuración de software) entre otros los que se detallaran a continuación.

2 Modelo de Proceso

Para el desarrollo del proyecto se ha optado por un modelo iterativo e incremental con una etapa previa de definición e investigación y una etapa posterior para ajustes y control. En la segunda etapa (iterativa-incremental) se realizaran tres iteraciones y de esta manera poder obtener una retroalimentación constante.

Dicho modelo fue elegido no solo porque ya ha sido utilizado por los integrantes del equipo a lo largo de la carrera sino que además permite la realización de prototipos evolutivos testeados y entregables que irán incrementando las funcionalidades de Gsiam en cada iteración, sumado a la flexibilidad que otorga a la hora de definir requerimientos así como también frente a posibles cambios de diseño.

2.1 Etapas

El proyecto se dividió en tres etapas:

Primer Etapa
Se dedicara mayormente a la investigación de las tecnologías a utilizar, investigación del mercado y productos similares. Dentro de la investigación se deberá realizar una prueba de concepto (POC) para determinar la factibilidad técnica de la realización del proyecto. También se definirán los requerimientos y la arquitectura de la aplicación. Otros de los hitos de esta son el plan de calidad del proyecto y también un análisis de riesgo que implica la realización de un proyecto de estas características.

Segunda etapa
Esta etapa se centra en el desarrollo del software propiamente dicho. A su vez se dividió en tres fases, de esta forma se llegara al software final de una manera incremental y controlada. Cabe destacar que cada fase se descompone en análisis, diseño, codificación, pruebas y ajustes Las mismas se detallan a continuación:

Fase1: Desarrollo CU prioridad alta

En esta fase se desarrollaran los casos de uso que se consideran más importantes para el funcionamiento del prototipo. Dentro de estas funcionalidades se destacan:

- Login
- Crear Usuario
- Ver Perfil de Usuario
- Buscar Usuario
- Buscar Sitio de Interés
- Crear Sitio
- Ver Sitio

Como se puede ver en esta fase se desarrollara el grueso de los casos de uso dejando para las etapas siguientes funcionalidades menores y ajustes.

Fase2: Desarrollo CU prioridad media
Esta fase abarca los casos de uso de prioridad media en cuanto a importancia para el sistema. Dentro de estas funcionalidades se encuentran:

- Ver Amigos
- Ver Solicitudes de Amistad
- Generar Solicitud de Amistad
- Responder Solicitud de Amistad
- Invitar Amigos
- Publicar Información

Fase3: Desarrollo CU prioridad baja
En esta última fase se pretender abarcar los casos de uso de prioridades bajas o no fundamentales para el correcto funcionamiento del sistema. Dentro de esos casos de uso se encuentran:

- Modificar-Eliminar Usuario
- Ubicar Amigos-Radar
- Compartir Ubicación
- Modificar-Eliminar Sitio

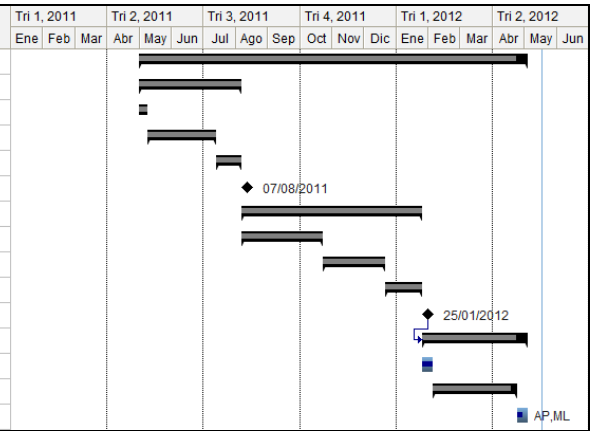
Tercer Etapa
La última etapa del proyecto está destinada a generar parte de la documentación (manuales, carpeta principal y presentación) y realizar las pruebas y ajustes finales.

3 Plan de proyecto

3.1 Gantt Resumido

A continuación se muestra el diagrama de Gantt resumido, para ver el detalle referirse al punto 1 de los anexos.

	Nombre	Duración	Inicio	Fin
1	**G-SIAM**	**368d**	**02/05/2011**	**04/05/2012**
2	**Etapa 1**	**97d**	**02/05/2011**	**07/08/2011**
3	**Fase 1: Configuración**	**8d**	**02/05/2011**	**10/05/2011**
8	**Fase 2: Investigación**	**65d**	**10/05/2011**	**14/07/2011**
21	**Fase 3: Elaboración**	**24d**	**14/07/2011**	**07/08/2011**
28	Fin Etapa 1	0d	07/08/2011	07/08/2011
29	**Etapa 2**	**171d**	**07/08/2011**	**25/01/2012**
30	**Iteración 1: Desarrollo de CUs prioridad alta**	**77d**	**07/08/2011**	**23/10/2011**
36	**Iteración 2: Desarrollo de CUs prioridad media**	**59d**	**23/10/2011**	**21/12/2011**
42	**Iteración 3: Desarrollo de CUs prioridad baja**	**35d**	**21/12/2011**	**25/01/2012**
48	Fin Etapa 2	0d	25/01/2012	25/01/2012
49	**Etapa 3**	**100d**	**25/01/2012**	**04/05/2012**
50	Pruebas y ajustes finales	10d	25/01/2012	04/02/2012
51	**Documentación**	**80d**	**04/02/2012**	**24/04/2012**
55	Presentación	10d	24/04/2012	04/05/2012



4 Análisis de Riesgo

4.1 Introducción

Como parte del área de gestión del proyecto, la identificación y estudio de los riesgos es fundamental a la hora de planificar acciones de contingencia que permitan minimizar efectos negativos.
Dada la existencia de fechas de entrega y la inexistencia de tiempo para corregir el daño de la concreción de riesgos con alta probabilidad, se considera importante adoptar una estrategia de control de riesgos proactiva. Por lo tanto el presente análisis se centra en la anticipación al riesgo, mediante la confección de una tabla de riesgos y el impacto que puede generar cada riesgo en particular.

4.2 Metodología y Matriz de Riesgos

En la matriz se pondera cada factor de riesgo en base a su probabilidad de ocurrencia e impacto. La probabilidad y el impacto se valoran en función a una escala de 1 a 10, correspondiendo el 1 a “bajo” y el 10 a “alto”. La Ponderación se calcula como probabilidad * impacto, siendo está valorada en función una escala de 1 a 100. A continuación se presenta la matriz resultante ordenada por prioridad:

N°	Riesgo	Probabilidad	Impacto	Ponderación
1	Inexperiencia del equipo en tecnologías utilizadas	8	8	64
2	Inadecuado conocimiento de los sistemas a integrarse	8	8	64
3	Errores de estimación en el calendario	7	7	49
4	El equipo no dispone de tiempo suficiente para el proyecto	6	8	48
5	Errores de diseño que dificulten la implementación	5	9	45
6	Falta de madurez en Frameworks y herramientas	4	8	32
7	Completitud y correctitud de los Requerimientos	4	7	28
8	Problemas de coordinación y/o comunicación del equipo debido a la distancia	4	5	20
9	Imposibilidad de concretar las reuniones de tutoría por tema de horarios de integrantes.	4	5	20
10	Pérdida de archivos por problemas de versionado	4	4	16
11	Falla en el Repositorio on line utilizado para almacenar la línea base	3	4	12
12	El tutor o algún integrante del equipo abandona el proyecto	1	9	9

4.3 Plan de Mitigación y Contingencias

En función de los riesgos detectados, se confecciona una tabla con planes de contingencia y medidas para mitigar cada riesgo.
Seguidamente se presenta dicha tabla:

Nro	Riesgo	Estrategia de Mitigación
1	Inexperiencia del equipo en tecnologías utilizadas	Se define realizar una investigación profunda de las tecnologías a utilizar en etapas tempranas del proyecto. El equipo debe apoyarse fuertemente en la experiencia técnica del tutor para evitar desvíos por inexperiencia.
2	Inadecuado conocimiento de los sistemas a integrarse	Se define realizar un estudio acerca de los sistemas a integrarse en las primeras etapas del proyecto. También se cuenta con la experiencia del tutor en estos sistemas.
3	Errores de estimación en el calendario	Si el proyecto se pasa más de los 8 meses permitidos, se presentará una solicitud de prórroga, mecanismo contemplado formalmente para los proyectos de grado del I.U.A.S.
4	El equipo no dispone de tiempo suficiente para el proyecto	Definición de un compromiso de horas de dedicación semanales mínimas para cada integrante. Seguimiento del avance por parte del equipo y el tutor.
5	Errores de diseño que dificulten la implementación	Se deberá pensar cuidadosamente la arquitectura del producto.
6	Falta de madurez en Frameworks y herramientas	Se documentará en detalle la arquitectura y el diseño, utilizando UML
7	Completitud y correctitud de los Requerimientos	Se validará el diseño y arquitectura con la colaboración del tutor
8	Problemas de coordinación y/o comunicación del equipo debido a la distancia	Se cuentan con varias herramientas de comunicación (mail, chat, teléfono, video llamadas, etc) para poder acortar las distancias. Además los integrantes del equipo se entienden muy bien, los mismos han trabajado conjuntamente a lo largo de toda la carrera universitaria y secundaria cosechando buenos resultados.
9	Imposibilidad de concretar las	Buscar una frecuencia de reuniones y un horario en donde los integrantes cuenten con disponibilidad. En

	reuniones de tutoría por tema de horarios de integrantes.	caso de que alguno no pueda asistir, se cuentan con buenas herramientas de comunicación ya probadas. También se pueden asignan tareas a ser realizadas para la próxima reunión.
12	Pérdida de archivos por problemas de versionado	Disponer siempre de la última versión local como buena práctica, además se generarán respaldos de los fuentes en forma periódica (plan de Gestión de la Configuración)
11	Falla en el Repositorio on line utilizado para almacenar la línea base	Disponer siempre de la última versión local como buena práctica, en caso de fallo solo se debe sincronizar los últimos cambios realizados por cada integrante. Además se cuenta con los respaldos de forma periódica (plan de Gestión de la Configuración)
12	El tutor o algún integrante del equipo abandona el proyecto	La probabilidad de ocurrencia es muy baja, dada la motivación del equipo y del tutor

5 Plan de Pruebas

5.1 Introducción

5.1.1 Misión

El propósito de las pruebas es corroborar que los requerimientos planteados en el documento de especificación de requerimientos (ERS) son cubiertos por el sistema.
El plan de pruebas tiene como misión definir las pautas para poder cumplir con el propósito de las pruebas de manera exitosa.

5.1.2 Objetivo

El objetivo de este plan de prueba es documentar la estrategia y el enfoque de todas las fases de prueba correspondientes al proyecto de grado G-SIAM, a fin de garantizar el cumplimiento de la misión especificada anteriormente.
Mediante este plan se especifica el tipo de pruebas que se realizan, para verificar el comportamiento esperado del sistema, y la adecuación a los estándares de calidad

definidos tanto para el sistema como para la documentación. Se define también cuando y como se realizarán.

Lo que se busca al ejecutar distintos tipos de pruebas de forma correcta es:

- Identificación temprana de la mayor cantidad de defectos, con el fin de reducir costos.
- Encontrar problemas importantes para minimizar riesgos.
- Verificar que el sistema cumple con el ERS.
- Dejar constancia de errores y problemas mediante documentación.
- Constatar la adecuación a estándares.
- Comprobar la calidad del sistema.

5.2 Tipos de Pruebas

Dadas las características del proyecto el Equipo define realizar tres tipos de pruebas:

- pruebas unitarias de los servicios (Caja Negra). Mediante JUnit.
- pruebas funcionales (Caja negra). Se realizarán pruebas sobre las interfaces finales del sistema para comprobar el funcionamiento esperado.
- Se contará con pruebas relacionadas a la documentación y el código.

5.2.1 Pruebas unitarias de los servicios

Para automatizar las pruebas de los servicios se realizarán pruebas unitarias de estos con el framework Junit junto a una librería para facilitar el testeo de servicios REST llamada REST-assured. Las pruebas se ejecutan desde el código fuente.

5.2.2 Pruebas Funcionales

El objetivo de las pruebas funcionales o de caja negra es verificar que el sistema satisface todos los requerimientos (ERS) y funciona según lo especificado en los Casos de Uso.
Básicamente el enfoque de este tipo de prueba se basa en el análisis de los datos de entrada y en los de salida de una funcionalidad en particular.

5.2.3 Pruebas sobre la documentación

La finalidad de estas pruebas es verificar que la documentación y el código generado cumplan con los estándares definidos.

5.3 Estrategia de pruebas

5.3.1 Metodología de Prueba

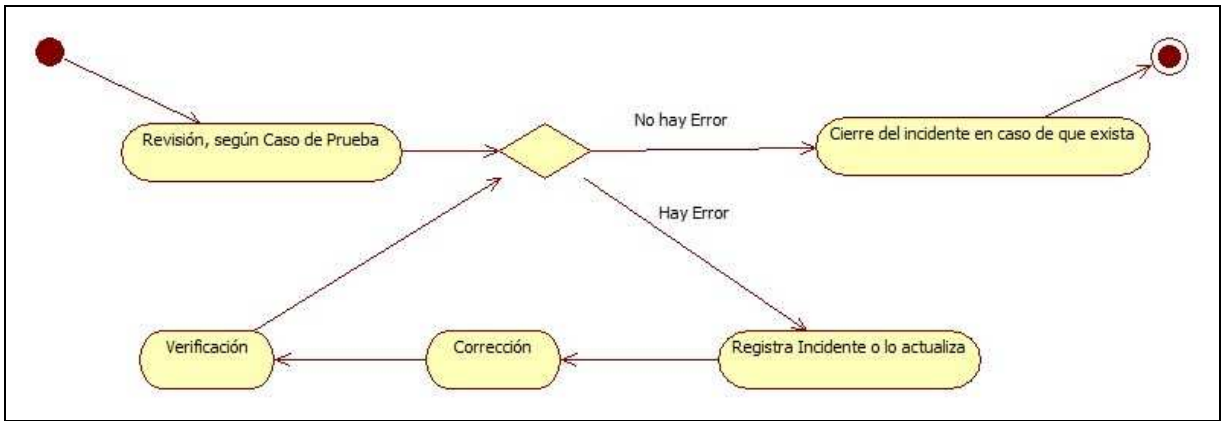
Las pruebas de Caja Negra las realiza el responsable de testing cuando el programador realiza una entrega (parcial o total) del programa. Dado el tamaño del proyecto estas pruebas serán cruzadas, es decir, al ser solo dos integrantes, la persona que construya un CU testeara el CU de la 2da persona y viceversa.
Las pruebas funcionales o de Caja Negra estarán basadas en casos de pruebas.

Las pruebas sobre la documentación podrán ejecutarse opcionalmente al finalizar cada iteración, pero si será obligatorio realizar una prueba completa antes de finalizar el proyecto.

5.3.2 Proceso y ciclos de pruebas del proyecto

El primer paso de este proceso será diseñar los casos de pruebas definidos para el proyecto (ver punto 5.3.3), luego se tendrán que preparar los datos de prueba, con esto ya se podrá ejecutar la aplicación para posteriormente comparar los resultados obtenidos con los esperados escritos en los casos de prueba. Al terminar la ejecución se debe documentar apropiadamente en la ficha si la prueba se da como exitosa o no, en caso de que se registre un error, se deberá crear una incidencia en el sistema de tickets (Issue Tracking de GoogleCode) [*ref.003 y ref.004*]. Una vez corregidos los errores, se debe ejecutar nuevamente la prueba y registrar los resultados. Si hay éxito, se ejecutan los cambios, en caso negativo nuevamente deben documentarse los errores.

A continuación se representa gráficamente el proceso:



5.3.3 Casos de Prueba

A continuación se muestra un caso de prueba a modo de ejemplo:

ID CU	Nombre	ID CP	Requisitos	Descripción del caso de prueba	Datos de prueba	Valor esperado	Resultado (ok, error)	Observaciones	Responsable	Validación
CU01	Ingreso del usuario	CP01-01	Ya existe el usuario gsiam.test@gmail.com en el sistema (CP02-01)	El usuario ingresa datos correctos.	**e-mail:** gsiam.test@gmail.com **contraseña:** 123	El usuario ingresa a la aplicación y se muestra la pantalla principal junto a su nombre (Gsiam Test).	OK		Martin Loureiro	

ID CU: Identificador de caso de uso al cual se le ejecutara el caso de prueba.
Nombre: Nombre del caso de prueba.
ID CP: Identificador único del caso de prueba.
Requisitos: Precondiciones para la correcta ejecución del caso de prueba.
Descripción del caso de prueba: Breve comentario de lo que debe hacer el caso de prueba.
Datos de prueba: Datos para poder ejecutar el caso de prueba.
Valor esperado: Valor correcto esperado después de la ejecución del caso de prueba.
Resultado: Resultado correcto (ok) no correcto (error).
Observaciones: Comentarios u observaciones del caso de prueba.
Responsable: Ejecutor del caso de prueba.
Validación: Validación de caso de prueba.

NOTA: Para ver el juego completo de casos de prueba referirse al punto 12 de los anexos.

5.4 Herramientas

5.4.1 Registro y Seguimiento de defectos

Como se mencionó previamente se deben crear incidentes con los defectos encontrados. Para esto se utiliza el sistema de manejo de tickets que viene incorporado al Google Code llamado Issue Tracking. El mismo contiene un título donde debe describirse el problema en pocas palabras. Luego se detalla el problema lo mejor posible, se pueden adjuntar archivos. Al ticket se le asigna una prioridad, un estado, el tipo de defecto, un mail de copia y se indica a que módulo del sistema afecta.
Los posibles estados son: New, Accepted, Started, Fixed, Verified, Invalid, Duplicate, WontFix y Done.
Las posibles prioridades son: Critical, High, Medium, Low.

A continuación se muestran imágenes de la herramienta:

6 Plan de SQA

6.1 Introducción

El objetivo del Plan de de aseguramiento de la calidad es establecer por escrito los objetivos, métodos y responsabilidades asociados a la calidad del software.

El plan de aseguramiento de la calidad que utiliza el equipo de proyecto, se basa en el resumen de la norma IEEE Std-983-1986 [*ref.005*], visto en la asignatura “Ingeniería de Software II” en el tercer año de la carrera de “Licenciatura en Informática”.

6.2 Secciones

6.2.1 Sección 1 – Propósito

En el presente Plan de SQA se describe la estrategia para el aseguramiento de la calidad durante el proyecto. El objetivo de este plan es garantizar un estándar de

calidad mínimo, tanto proceso de construcción del software “G-SIAM”, como en el producto final, entendiéndose por producto el software más su documentación.

6.2.2 Sección 2 – Documentos de referencia

Norma IEEE Std-1986.

6.2.3 Sección 3 – Gestión

Existe un rol de encargado de las actividades de SQA, llamado “Encargado de SQA” que no participa en tareas de desarrollo. El Encargado de SQA es responsable de generar la documentación correspondiente así como de redactar el Plan, y hacer directamente las tareas de control y seguimiento.

Ya que el equipo de trabajo es de sólo 2 personas y no se dispone de un recurso externo que pueda realizar estas tareas, las mismas serán realizadas por alguna de las personas del equipo.

6.2.4 Sección 4 – Documentación

La documentación del proyecto consiste en un conjunto principal de documentos denominados carpeta principal, y otro grupo de documentos complementarios denominados anexos. En la carpeta principal se encuentra el plan de proyecto, el documento de análisis de riesgo, el plan de pruebas, el presente plan de SQA, el plan de SCM, el documento de especificación de requerimientos, la documentación de arquitectura, documento de estado del arte entre otros documentos de alta relevancia. En los anexos se incluyen casos de prueba, manual de instalación, manual de usuario, documento de investigación de mercado y demás información complementaria.

6.2.5 Sección 5 – Estándares, Prácticas y Convenciones

Los procedimientos de gestión de configuraciones se pueden observar en detalle en el Plan de Gestión de Configuración de Software (ver punto 7). Las prácticas y procedimientos del aseguramiento de calidad están identificadas en el propio Plan de Calidad.
Mediante revisiones se controla que todas las prácticas y los estándares sean cumplidos.

6.2.5.1 Estándar de Documentación

A continuación se describe el estándar definido para este proyecto.

- Tipo de hoja A4.
- Tipo de letra Arial.
- Tamaño de letra 12.
- Párrafos justificados.
- Los párrafos son separados por un renglón en blanco.
- Títulos 4 niveles.
- Letra de los títulos Arial.
    - Título 1 tamaño 20 negrita.
    - Título 2 tamaño 18 negrita y cursiva.
    - Título 3 tamaño 16 negrita y cursiva.
    - Título 4 tamaño 14 negrita y cursiva.
    - Título 5 tamaño 12 negrita y cursiva.
- Para la carpeta se utilizaran capítulos (Arial tamaño 22), los mismos deberán comenzar en una página exclusiva.
- Si un titulo queda en la última línea de la hoja, el titulo deberá pasar a la siguiente hoja.

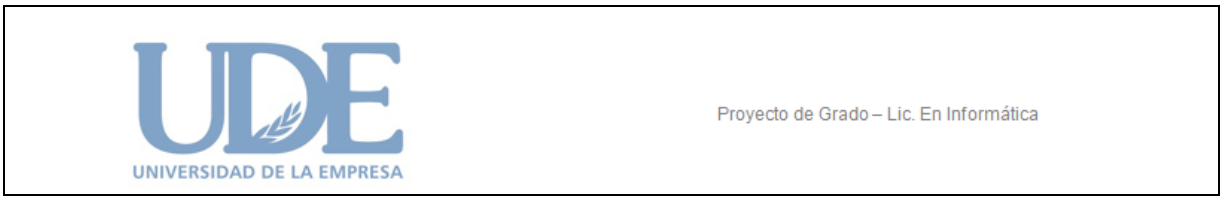
- Las imágenes irán centradas y con un borde negro de ½ punto.

- Para referirse a segmentos de código se utilizara el siguiente contenedor:

```
SELECT * FROM DUAL
```

- Todos los documentos contendrán una carátula como página principal, la misma se confecciona en 3 partes:

Cabezal



Principal

Geolocalización de Sitios de Interés Para Aplicaciones Móviles

G-SIAM

Nombre del Documento

VERSIÓN 1.0

Pie

Universidad de la Empresa
Soriano 959 - Tel.: 2900 2442
Montevideo - Uruguay | info@ude.edu.uy

- Todas las paginas menos la carátula tendrán un cabezal y un pie de página:

Cabezal

Geolocalización de Sitios de Interés Para Aplicaciones Móviles G-SIAM
Nombre del Documento Versión: 1.0

Pie

- Los documentos al referirse al equipo de trabajo utilizan únicamente las siguientes opciones
  - El equipo
  - El equipo de proyecto
  - Omitiendo el sujeto hablando en tercera persona del singular

- Se adjuntan algunos ejemplos que indican el modo en que se expresan los documentos
  - "El equipo decide desarrollar un modelo incremental iterativo"
  - "Se decide desarrollar un modelo incremental iterativo"

- Como buena práctica se creó un template de documento para facilitar la creación de nuevos documentos, el mismo se puede ver en punto 17 de los anexos.

6.2.5.2 Estándar de Código

En cuanto a la codificación, se utiliza el estándar oficial de codificación del lenguaje Java [*ref.006*], con la excepción de las salvedades que se listan más adelante. Este garantiza una mejor comprensión y mayor facilidad de revisión de programas, en un ambiente que involucra a más de un programador. Esto garantiza la calidad del código generado por cada uno de los miembros del equipo de trabajo.

Los siguientes elementos puntuales se admiten y priman sobre la definición del estándar antes mencionado.

- Se admiten tabuladores de 4 espacios
- Se admiten el uso del comentario de línea (//) para varias líneas sucesivas, a modo de comentario de bloque.
- Para la estructura de control “If else - if” además del formato mencionado en el estándar se admite el siguiente:

```
If (condición) {
                    //sentencias
 }
  else
        If (condición){
                //sentencias
          }
```

- Los nombres de los métodos no sólo pueden ser verbos en infinitivo, sino que también se pueden usar otras palabras del idioma español. Se persigue el objetivo de que se entienda claramente el sentido del método y se respete la regla de mayúsculas y minúsculas descrita en el estándar.

6.2.6 Sección 6 – Revisiones

Se define que se realizarán Revisiones Técnico Formales (RTF) al final de cada iteración. Ambos integrantes deberán controlar que el código fuente y los documentos generados se ajusten a los estándares definidos en los documentos de referencia.

6.2.7 Sección 7 – Gestión de Configuración de Software

Se dispone de un plan de SCM que detalla las actividades de ésta área. (Ver punto 7, “Plan de Gestión de Configuración de Software”).

6.2.8 Sección 8 – Informe de problemas y acción correctiva

Los problemas detectados se informan a través del Issue Tracking existente en el sitio del proyecto. Esta tarea le corresponde al encargado de calidad. Adicionalmente en reuniones a las que convoca el encargado de calidad cuando cree conveniente, se realiza una devolución de los problemas más generales encontrados con el objetivo de que otros miembros del equipo no cometan los mismos errores. Dado el tamaño del equipo completo de proyecto no se cree conveniente prefijar las fechas de estas reuniones.

6.2.9 Sección 9 – Herramientas

6.2.9.1 Issue Tracking

Esta herramienta dispone de 3 tipos de templates:

- Reporte de defecto del usuario
- Reporte de defecto del desarrollador
- Solicitud de revisión

Además cuenta con varios estados para los problemas reportados y son configurables, es decir se pueden agregar más según necesidad.

6.2.9.2 Metrics

Para la generación de métricas de código se utiliza el plugin de eclipse 'metrics' [*ref.007*], el cual analiza el código y presenta diferentes métricas. Esta herramienta permite obtener parámetros útiles para realizar un seguimiento del desarrollo. La definición completa de las métricas ofrecidas por la herramienta se puede consultar en el sitio.

6.2.9.3 Wiki

Para tener la información del proyecto on line se utiliza la wiki [*ref.008*] que viene incluida dentro del sitio del proyecto (Google Code). En ella se incluyen:

- Herramientas
- Tutoriales
- Consejos para desarrollar en Android
- Estándares
- Glosario

Para ver imágenes, ir al punto 16 de los anexos.

6.2.10 Sección 10 – Control de Código

Todo lo que es versionado del código se gestiona como se explica en el Plan de SCM (ver punto 7).

6.2.11 Sección 11 – Control de Medios

En el plan de SCM se detalla el control que se tiene sobre los medios (ver punto 7).

6.2.12 Sección 12 – Control del Proveedor

No aplica a este proyecto ya que no hay proveedores.

6.2.13 Sección 13 – Registro, Mantenimiento y Retención

El equipo ha decidido almacenar toda la documentación del proyecto sin destruir nada en ningún momento. El objeto de esto es tener acceso a todos los datos históricos en el

momento que se desee. Para ello se cuenta con herramientas de versionado apropiadas, y el espacio de almacenamiento no supone un problema.

6.3 Métricas

Se decide obtener métricas en base al código generado y en base al esfuerzo requerido por cada tarea. Para las métricas de código las mediciones se realizaran luego de cada cierre de cada iteración. Mientras que para las de esfuerzo se realizaran al final de cada etapa y cada Iteración.

6.3.1 Métricas de Código

- Líneas de Código (TLOC y MLOC)
- Promedio de Métodos por Clase (NOM / NOC)
- Complejidad Ciclomática McCabe (CC)
- Profundidad del Árbol de Herencia (DIT)

A través de las 2 primeras métricas (Líneas de Código y Promedio de Métodos por Clase) se pretende tener cuantificado el tamaño del desarrollo. Mientras que las últimas 2 (Complejidad Ciclomática McCabe y Profundidad del Árbol de Herencia) intentan medir la calidad de la lógica del programa generado, y el riesgo que insume su desarrollo y mantenimiento.
Es importante realizar este seguimiento para poder detectar las áreas que insumen mayor esfuerzo de desarrollo y por consiguiente mayores pruebas.

6.3.2 Métricas de Esfuerzo

- Esfuerzo por Actividad
- Incidencia de la Actividad (horas de la actividad / horas totales de la iteración)
- Esfuerzo por CU

Con estas métricas se obtienen datos relevantes que indican cómo se divide el esfuerzo dentro del proyecto.

Estas métricas permiten comenzar a generar datos útiles a fin de poder lograr una estimación más certera de futuros proyectos.
De este modo se podrá asignar con más exactitud los recursos y mejorar sus costos.

7 Plan de Gestión de Configuración de Software

7.1 Objetivo

El plan de gestión de configuración de software describe la metodología a aplicar para gestionar el software y la documentación del proyecto. Dicha metodología define, los pasos a seguir para asegurar la integridad, disponibilidad y versionado del código fuente así como de la documentación de todo el proyecto.
Con este plan se pretende suavizar el impacto que causan los cambios no controlados y resolver los problemas de actualización simultanea de código y documentación.

7.2 Organización

Dado que el equipo de proyecto es de dos integrantes, no se vio oportuno generar una estructura formal en lo que refiere a la aprobación de los cambios. Sin embargo los cambios de relevancia se discuten y aprueban en conjunto previamente a impactar los repositorios de código y documentación.

7.3 Métodos y Herramientas

Tanto para el manejo de versionado del código fuente como para la documentación se utilizaran herramientas Open Source [*ref.009*] provistas por Google.
Para lo que a documentación se refiere, se utilizara Google Docs, que es un servicio Could Computing [*ref.010*] que permite visualizar y editar documentos de texto, hojas de cálculo e imágenes entre otros tipos de archivos.
Además de poder trabajar en línea sobre cualquier documento del proyecto, también se puede manejar el versionado ya que esta herramienta permite volver a una versión anterior de cualquiera de los archivos de una forma fácil e intuitiva.
Se cree que dadas las condiciones en las que se están desarrollando el proyecto, una herramienta como esta será de gran ayuda ya que permite hacer modificaciones a un documento en forma simultánea y en tiempo real, viendo en todo momento que es lo que está modificando la otra persona.
Otras de las características que inclinaron al equipo para elegir a Google Docs como repositorio de documentos fue la facilidad con la que permite hacer respaldo de los archivos a disco.
En cuanto a repositorio de código se opto por el que provee Google Code, el cual es para proyectos Open Source. Dentro del mismo, se eligió el sistema de control de versiones Subversion [*ref.011*], debido a la experiencia que el equipo posee en el

desarrollo sobre esta herramienta. Cabe destacar que Google ofrece otros sistemas para manejar el versionado.
Además del versionado del código esta herramienta brinda una wiki donde incluir información relevante y de fácil acceso para los miembros del proyecto, un sistema para reportar incidentes o asignar tareas las cuales son notificadas por correo electrónico, también permite ver el código de forma on line, entre otras utilidades.
Para acceder al repositorio de código de una forma más fácil se utiliza un plugin que se instala en el entorno de desarrollo, el cual sincroniza nuestra copia local del código con la del servidor SVN.

7.4 Seguridad y Respaldo

Será responsabilidad de los miembros del equipo hacer un respaldo semanal a disco de todos los archivos. Para la documentación se utilizará la herramienta Open Source llamada Gdocbackup [*ref.012*] que permite descargar todos los archivos de Google Docs a un dispositivo local. Para automatizar este respaldo se usará una tarea programada de Windows con el programador de tareas para que corra este proceso una vez a la semana. Para el código fuente se harán los respaldos manuales en formato ZIP en el mismo dispositivo local.
En cuanto a seguridad como se trata de un proyecto Open Source cualquier persona podrá ver el código, pero en esta etapa del proyecto solo podrá ser modificado por los integrantes del equipo. Los miembros del equipo tendrán los permisos necesarios para realizar tareas administrativas sobre los repositorios como ser el borrado de alguna rama.

Capítulo III. Investigación

8 Investigación de mercado

Es importante para la realización de este proyecto identificar cuáles son los productos que lideran el mercado en cuanto a Geolocalización de sitios de interés ya que otorga una visión global de como se encuentra el mercado, que productos lo lideran y cuáles son sus ventajas y desventajas.

Para más detalles, referirse al punto 4 de los anexos.

9 Estado del Arte

9.1 Introducción

Esta sección pretende dar un cierre a la etapa de investigación de las tecnologías a utilizar en el proyecto, haciendo un análisis de las posibles alternativas en cuanto a sistemas operativos móviles existen en el mercado y por qué se eligió Android [*ref.013*] como plataforma a utilizar.
Ya que el proyecto cuenta también con una aplicación web, que es la que expone los servicios que consumirá el dispositivo móvil, se cree conveniente exponer en este documento las tecnologías investigadas para llevar a cabo esta aplicación.
Una de las características principales y más importantes del proyecto es de donde obtener los puntos de interés que se mostraran en el dispositivo móvil, se investigaron varias alternativas desde consumir servicios externos hasta implementar una base de datos propia de puntos de interés, analizando ventajas y desventajas de cada una.
Por último se analizaron las diferentes alternativas para la creación de la interfaz grafica.

9.2 Plataforma de Desarrollo

9.2.1 Java Micro Edition

La plataforma Java Micro Edition (JME) [*ref.014*] es un subconjunto de la plataforma Java para dispositivos con poca memoria y escasa capacidad de procesamiento como son PDA, electrodomésticos, teléfonos móviles entre otros.
En un entorno de ejecución JME aparecen diferentes componentes.

- Distintas maquinas virtuales (MV) dependiendo de la capacidad de procesamiento de cada dispositivo.
- Las configuraciones son el conjunto básico de librerías. Existen dos configuraciones:
  - CLDC (Connected Limited Device Configuration) enfocada a dispositivos con escasa capacidad de memoria como teléfonos móviles.
  - CDC (Connected Device Configuration) para dispositivos con mayor capacidad que la CLDC
- Los perfiles son librerías mas especificas de cada dispositivo orientadas a desarrollar funcionalidades de más alto nivel.

Al existir dos configuraciones diferentes cada una va a requerir su propia maquina virtual. La MV de la configuración CLDC se denomina KVM y para la configuración CDC se denomina CVM.
La Kilobyte virtual machine (KVM) adopta su nombre porque requiere muy poca memoria (entre 40Kb y 80Kb). Fue desarrollada por completo en C y con muy pocas líneas de código. Esta MV es la adecuada para dispositivos con muy poca capacidad de procesamiento.
La compact virtual machine (CVM) es una maquina virtual que posee todas las características de la MV estándar de Java y está pensada para dispositivos con una capacidad de procesamiento mayor como podrían ser electrodomésticos.
El objetivo principal de las configuraciones es el de garantizar la portabilidad entre diferentes dispositivos con escaza memoria y definir las funcionalidades generales. Las funciones mas especificas son definidas en los perfiles.
Un Midlet es una aplicación Java desarrollada sobre el perfil MIDP con la configuración CLDC.
Cabe aclarar que la capacidad de los dispositivos móviles ha ido en aumento y hoy por hoy existen dispositivos de gran capacidad, llamados Smartphone (teléfonos inteligentes) que permiten correr aplicaciones mucho más ricas en términos de capacidad de procesamiento e interfaz gráfica.
Estos dispositivos cuentan con un sistema operativo que los dota de características propias del fabricante perdiendo quizás homogeneidad entre los dispositivos de diferentes SO pero ganando en la calidad de las aplicaciones que corren en estos.

JME al no ser un sistema operativo para dispositivos móviles no puede dotar a los Smartphone de aplicaciones de gran calidad por lo que ha quedado rezagado en la lucha de este mercado que tanto está creciendo.
Por lo dicho anteriormente y dadas las características de la aplicación a desarrollar que requiere de una interfaz grafica atractiva y funcionalidades tales como GPS y una capacidad de procesamiento razonable se descarta la elección de JME como plataforma a desarrollar el proyecto.
A continuación se analizarán los principales sistemas operativos para dispositivos móviles y se profundizará en el estudio de Android que fue el elegido para desarrollar el proyecto de grado.

9.2.2 Sistemas Operativos Móviles

Para la elección del sistema operativo móvil (desde ahora SO) en el cual se desarrollara el proyecto, el equipo tuvo que investigar los distintos SO que están presentes en el mercado.
La lista a continuación abarca los principales SO hoy en día, cabe aclarar que la no presencia de Symbian en esta selección es por su unión con Windows y su inminente desaparición en un futuro no muy lejano.

9.2.2.1 iOS

iOS es el SO desarrollado por Apple para dotar a sus terminales iPhone y actualmente a las cada vez más popular iPad [*ref.015 y ref.016].*
Está basado en el kernel de Mac OS lo que lo dota de un SO robusto y una interfaz gráfica que se caracteriza por su fácil manejo.
La arquitectura del iOS está conformada por cuatro capas, el núcleo del sistema, la capa de servicios, la capa de comunicación y la capa de Cocoa Touch que es un conjunto de librerías para crear aplicaciones en iOS.
Una de las desventajas de iOS es que no soporta Java, en su lugar se usa un lenguaje de programación llamado Objective-C que es un súper conjunto de C orientado a objetos.
Pero la gran contra de este SO y en particular de los productos de Apple, es que son códigos cerrados y se debe contar con el hardware y software especifico para desarrollar para estos productos por eso fue que se descarto como plataforma a utilizar en el proyecto.

9.2.2.2 Windows Phone

Es el SO de Windows para dispositivos móviles, que anteriormente se llamaba Windows Mobile. Se basa en el sistema Windows CE [*ref.017*] y se caracteriza por sus similitudes con las versiones de escritorio para facilitar la tarea al usuario.
Este SO será según los expertos el que experimentará mayor crecimiento en cuota de mercado en el periodo 2011 – 2015 [*ref.018*].
Si bien no se opto por este SO para el proyecto por razones de tiempo y de que no es Open Source, es una alternativa a evaluar en un futuro si se quiere seguir con el desarrollo de este proyecto.
Otra de las razones por la cual no se elijo este SO es porque es el que menos cuota de mercado tiene actualmente. La nueva versión Windows Phone 7, la cual pretende competir con los SO de mayor peso en el campo de los dispositivos móviles como Android, iPhone, BlackBerry, esta aun muy verde en comparación a los mencionados anteriormente.

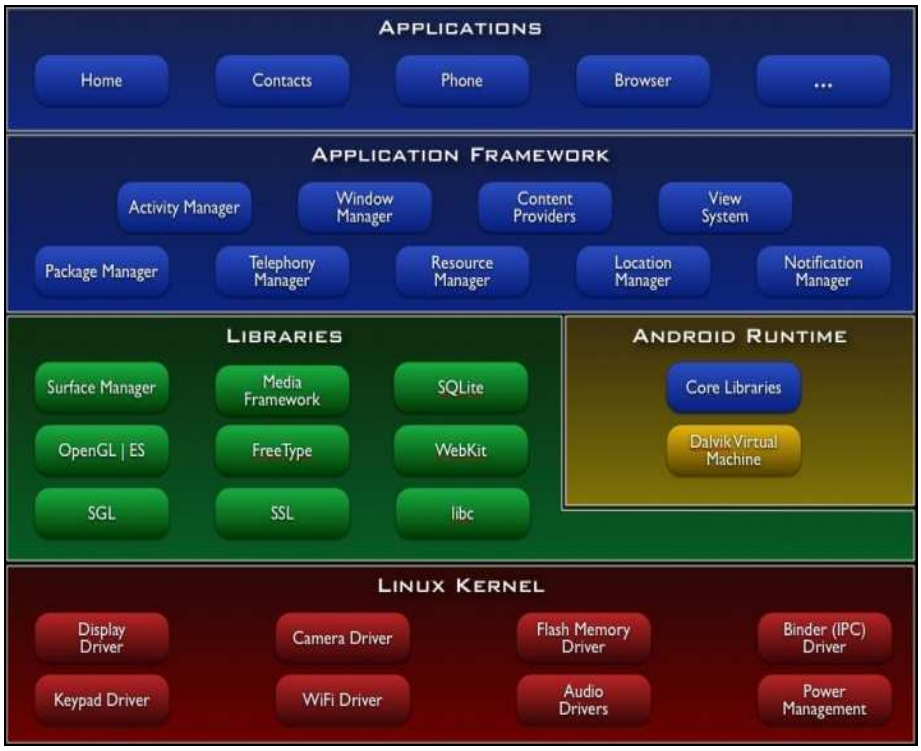
9.2.2.3 Android

Android es una plataforma abierta basada en el kernel de Linux 2.6 para dispositivos móviles desarrollado inicialmente por Android Inc y luego adquirido por Google. Actualmente se encuentra bajo la Open Handset Alliance que está compuesta por más de 50 empresas de desarrollo de hardware, software y operadores de servicios.
Esta diseñado en una arquitectura en capas donde el kernel de Linux hace de una capa de abstracción entre el hardware y el sistema.
En esta capa se encuentran los distintos servicios que ofrece esta plataforma como son la seguridad, gestión de memoria y procesos, manejo de drivers y protocolos de red.
Por encima de esta capa se encuentran el conjunto de librerías, que es la base para el desarrollo de aplicaciones sobre esta plataforma.
El runtime de Android es un conjunto de bibliotecas que determinan la base para las librerías escritas en Java.
Uno de los principales componentes del runtime es la Maquina Virtual de Dalvik. Esta permite que cada aplicación Android corra su propio proceso en una maquina virtual Dalvik completamente independiente de los demás procesos. Esta máquina virtual ejecuta archivos .dex los cuales son optimizados para un bajo consumo de memoria.
Cuando el código se compila se generan archivos .class que luego son transformados archivos .dex.
Al mismo nivel del runtime se encuentran un conjunto de librerías escritas en C/C++ que son accedidas mediante la capa de aplicación. Sus funcionalidades abarcan desde la persistencia de datos mediante SQLite pasando por librerías multimedia para audio y video y manejo de aplicaciones graficas 2D y 3D mediante OpenGL.
La siguiente capa es quizás la más importante para los programadores ya que es con la que tienen contacto directo para el desarrollo de aplicaciones Android. Esta capa es el framework de Android y es la que utiliza las características antes planteadas.

Esta capa contiene varios componentes:

- Un conjunto de vistas que permite desarrollar la interfaz gráfica.
- Los proveedores de contenidos son el método de intercambio de información entre aplicaciones.
- Administrador de recursos que permite acceder a recursos como strings, gráficos, archivos de layout, etc.
- El gestor de notificaciones permite generar alertas personalizadas al sistema.
- El gestor de aplicaciones es el encargado de manejar el ciclo de vida de las actividades.

La última capa, es la capa de aplicaciones que vienen por lo general ya instaladas en los sistemas operativos Android. Esta capa es desarrollada con todas las aplicaciones antes mencionadas e incluye cliente de correo, calendario, navegador, agenda de contactos entre otros.
A continuación se muestra la arquitectura de Android:



[*ref.019*]: Arquitectura Android

9.2.2.3.1 Funcionalidades Generales

Las aplicaciones en Android se construyen en base a bloques de software elementales. Aunque no es necesario que una aplicación los utilice todos [*ref.020*].
Activity: Una actividad es sin duda el componente más utilizado en una aplicación Android. Se puede definir como una tarea que se lleva a cabo en la aplicación y tiene una interacción con el usuario, en otras palabas se puede decir que una actividad es una pantalla de la aplicación. Las aplicaciones estarán compuestas por muchas actividades, que relacionadas entre sí y cada vez que se accede a una nueva actividad, la anterior es almacenada en una pila que gestiona el ciclo de vida de las actividades. Gracias a esto el usuario podrá navegar a una actividad anterior.

Intent: Un intento como dice su nombre es la intención por hacer algo como puede ser "abrir el navegador" o "enviar un SMS". Android por su parte buscara la aplicación más adecuada para llevar a cabo esta petición. Estos intentos pueden ser los predefinidos por Android como se menciono anteriormente o los creados por el programador para navegar entre distintas actividades.

Service: Un servicio es un proceso que se ejecuta en background sin necesidad de que haya interacción con el usuario y probablemente por un largo periodo de tiempo. Se podría decir que es equivalente al daemon de Unix. Por ejemplo un servicio podría estar ejecutando el reproductor de música, realizar operaciones de entrada/salida o enviar un SMS en background.

Content Provider: Los proveedores de contenido recuperan y almacenan la información haciéndola accesible para otra aplicación. Es la forma que Android tiene para compartir información ya que no hay una zona donde todos paquetes puedan acceder.

9.2.2.3.2 Ciclo de Vida de las Aplicaciones

Android maneja una jerarquía de procesos para determinar cuál debe continuar y cual debe matar en base a los componentes y el estado que se están ejecutando.

- **Proceso en primer plano:** Un proceso en este estado es aquel en el cual el usuario esta interactuando. Este será la última opción cuando se tenga que eliminar uno.

- **Proceso visible:** Este es un proceso que es visible por el usuario pero este no está interactuando directamente con él. Esta situación se puede dar por ejemplo cuando hay una confirmación en una ventana emergente. Al igual que el anterior, un proceso en este estado será de los últimos en ser elegidos para su eliminación por falta de memoria.

- **Proceso de servicio:** Aunque un proceso de servicio no es visible por el usuario y no necesitaría interacción de este, si hace cosas que al usuario le importa cómo puede ser el reproductor de música.

- **Proceso de segundo plano:** Estos procesos no se encuentran visibles por el usuario y el sistema podría matarlo en cualquier momento para recuperar memoria para otro proceso.

- **Proceso vacio:** Los procesos vacios son aquellos que no contienen ningún componente activo y son frecuentemente eliminados por el sistema. La única razón para mantenerlos activos es para manejar un cache y mejorar los tiempos de inicio.

9.2.2.3.3 Ciclo de Vida de las Actividades

El ciclo de vida de una actividad es una parte muy importante para desarrollar aplicaciones de alto rendimiento y flexibles. Las actividades se van almacenando en una pila de actividades, cuando se llama a una actividad esta pasa a la parte superior de la pila y la anterior queda por debajo de esta en background hasta que vuelva nuevamente a la parte superior o sea eliminada por falta de memoria.
Los estados de una actividad son:

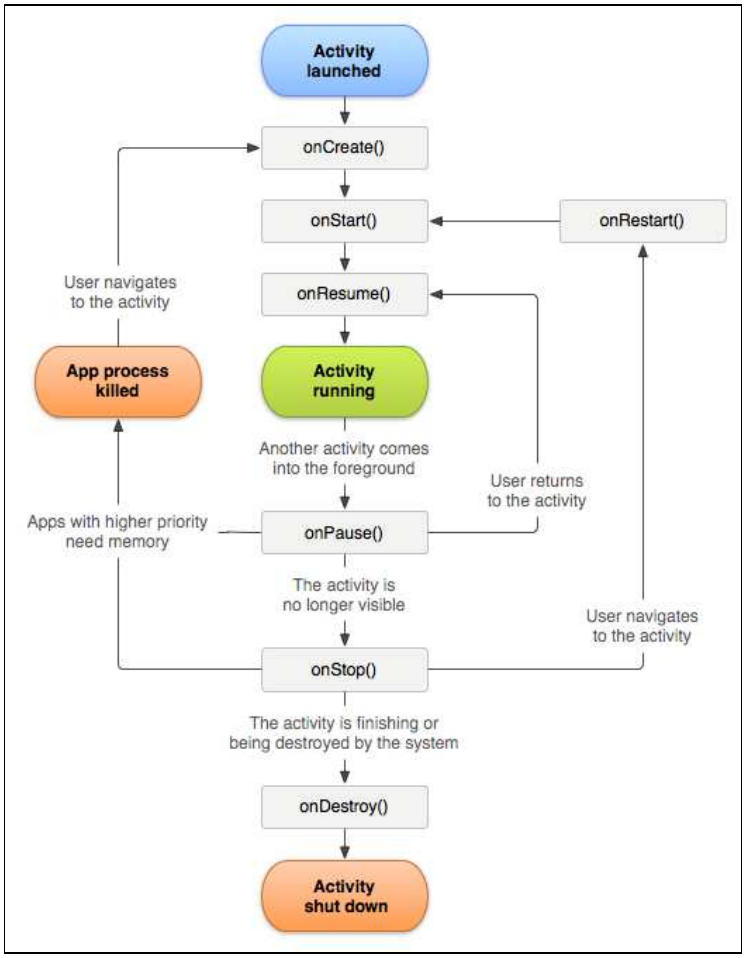
- **En ejecución:** La actividad se encuentra en el tope de la pila y tiene el foco del usuario.

- **En Pausa:** Una actividad está en pausa cuando deja de ser la principal pero aun es visible por el usuario. Esto es posible cuando se llama a una ventana emergente transparente de confirmación que deja ver la actividad anterior. Cuando la ventana emergente se cierra la anterior continúa con sus datos en el estado de ejecución.

- **Parada:** En este estado la actividad no es visible por el usuario pero sigue manteniendo toda la información. Sin embargo una actividad en este estado es elegible para ser eliminada de la pila si es que el sistema necesita memoria para otra aplicación.

El ciclo de vida de una aplicación está muy relacionado con el ciclo de vida de una actividad.
Android da prioridad a las actividades que están en interacción con el usuario. Para manejar el ciclo de vida de una actividad en Android existen los métodos de la clase Activity que son invocados con los cambios de estados correspondientes. Los métodos son: OnCreate(), OnDestroy(), OnPause(), OnStop(), OnFrezze(), OnResume() , OnRestart() y Finish().

A continuación se muestra un diagrama del ciclo de vida de una Activity:



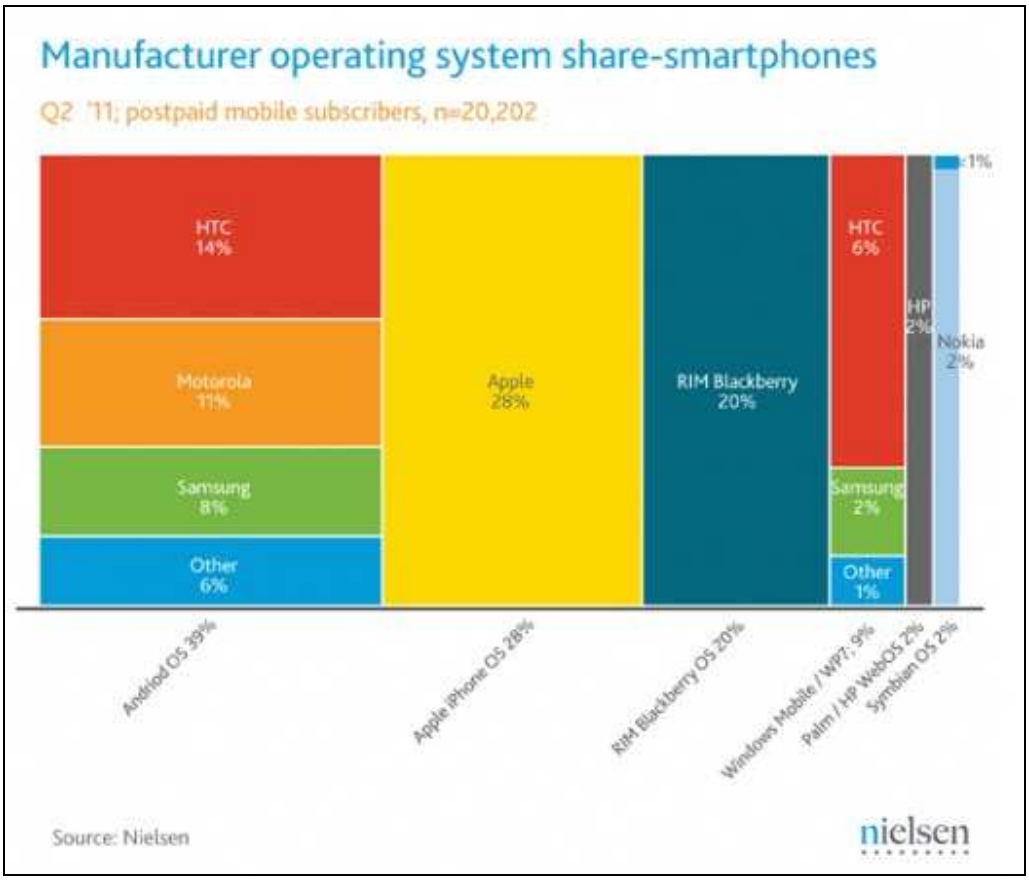
[*ref.021*]: Ciclo de vida de una Activity

9.2.3 Conclusiones

La elección de Android como plataforma en la cual desarrollar el proyecto se debe básicamente a tres puntos

- Es Open Source, cumpliendo uno de los requisitos planteados del proyecto.
- Además de ser el líder en cuota de mercado de hoy en día, las proyecciones lo dan como el ganador a futuro con el 43.8% de mercado en el 2015 [*ref.022*].
- Por último dado la experiencia de los integrantes del equipo en el lenguaje de programación Java y dado que Android utiliza este lenguaje es otro de los puntos a favor que determinaron en la elección de esta plataforma.

La siguiente infografía muestra la cuota de mercado de los sistemas operativos para dispositivos móviles más importantes.



[*ref.023*]: Android Estadísticas

9.3 Obtención de Puntos de Interés

9.3.1 Introducción

La obtención de puntos de interés es uno de los pilares fundamentales del proyecto y uno de los que genero mayor investigación.
Un punto de interés desde ahora en más PDI, se entiende como cualquier sitio físico que pueda ser localizado en un mapa por un par de coordenadas geográficas y resulte de utilidad para el usuario.
Se analizaron distintas formas de obtener dichos puntos, desde servicios web de terceros hasta la creación de una base de datos propia con los PDI.

A continuación se detallan las diferentes alternativas investigadas.

9.3.2 Servicios Web de Terceros

Una forma de obtener los PDI es consumiendo servicios de terceros mediante un llamado por RESTful al recurso que se quiere obtener. Dentro de los proveedores más importantes se encuentra Simplegeo y Geonames.
Ambos son bases de datos con millones de PDI que pueden ser consultados por llamadas a los servicios que exponen.

9.3.2.1 SimpleGeo

Simplegeo es una base de datos con millones de PDI que expone mediante servicios web, además ofrece apis de desarrollo para diferentes lenguajes de programación como Java, Object-C, Java Script, Python lo que la convierte en una alternativa interesante [*ref.024*].
Simplegeo expone tres tipos de productos:

- Simplegeo Place está basado principalmente en la obtención de PDI.
- Simplegeo Context obtiene datos del contexto en el cual nos encontramos (temperatura, datos del clima, etc).
- Simplegeo Storage permite almacenar información en la nube de forma rápida y en tiempo real.

9.3.2.1.1 Ventajas

Dentro de las ventajas que se encontraron se debe destacar que es una base de datos con millones de registros de rápido acceso y robustez.
Permite consumir servicios web con diferentes criterios de búsqueda como puede ser PDI cercados a la ubicación del dispositivo a un radio dado.
Otra de las ventajas es que permite ingresar nuevos PDI, esto es de gran importancia para el proyecto ya que uno de los requerimientos es el de ingresar nuevos sitios, además de que pose muy pocos PDI para Uruguay.

9.3.2.1.2 Desventajas

La gran desventaja que se encontró es que no es un servicio del todo gratis ya que limitan el número de llamadas por día, cosa que para esta instancia no es de mayor importancia pero si se quiere continuar con este proyecto es un tema a tener presente.
Otra de las grandes desventajas al consumir servicios de terceros es la disponibilidad de los mismos y que la aplicación queda atada al buen funcionamiento de estos.

9.3.2.2 Geonames

Geonames es otra base de datos con millones de registros PDI al igual que Simplegeo, también expone los recursos mediante servicios web con un alto grado de disponibilidad y robustez.
También ofrece una gran cantidad de librerías escritas en diferentes lenguajes de programación como son Java, Objective-C, Ruby, Python, entre otras para acceder a estos servicios [*ref.025*].

9.3.2.2.1 Ventajas

Una de las ventajas de utilizar Geonames es que es una base de datos con millones de PDI que se pueden consultar mediante servicios web con diferentes clientes.

9.3.2.2.2 Desventajas

La gran desventaja que se encontró en esta alternativa es que no permite por código ingresar nuevos PDI y este es un requerimiento importante dentro del proyecto.

9.3.3 Base de datos Propia

Otra de las alternativas que se investigo y por la cual se opto es la de tener una base de datos propia donde guardar los PDI.
Para poder manejar estos datos de una forma rápida y eficiente es necesario manejar datos espaciales [*ref.026*] para saber la posición geográfica del PDI. Se opto por usar PostgreSQL [*ref.027*] con un plugin llamado Postgis [*ref.028*] para el manejo de datos geográficos.

9.3.3.1 Funcionalidades generales

PostgreSQL es una base de datos Open Source que cuenta con una gran cantidad de desarrolladores. El hecho de que sea Open Source es de gran importancia para el proyecto ya que este es un pilar importante del este.
Postgis es un plugin de PostgreSQL que permite manejar y realizar consultas sobre datos espaciales (GIS). Con Postgis se pueden manejar todos los objetos definidos en la especificación OpenGIS como son puntos, líneas, polígonos, multipuntos, etc. En el proyecto solo se manejaran puntos para representar a los sitios de interés pero cabe la posibilidad de utilizar otros objetos en futuros trabajos.
Los datos espaciales son guardados en la base de datos en una columna de tipo GEOMETRY que es agregada a PostgreSQL al momento de instalar el plugin.

A continuación se muestran ejemplos de consultas SQL para el manejo de datos espaciales:

```
INSERT INTO SPATIALDATABASE(THE_GEOM,THE_NAME)
VALUES(GeometryFromText('POINT(-126.4 45.32)',312),'Un Lugar')
```

Esta consulta es un ejemplo de cómo insertar un punto en una tabla de la base de datos que permite manejar datos GIS

```
SELECT *
FROM GEOTABLE
WHERE
GEOM && GeometryFromText('BOX3D(900 900,1100 1100)',-1)
AND
Distance(GeometryFromText('POINT(1000 1000)',-1),GEOM)<100;
```

Esta consulta muestra como seria la forma de recuperar los objetos que se encuentran a menos de 100 metros de distancia del punto POINT (1000, 1000).

9.3.3.2 Ventajas

Entre sus principales ventajas se encuentran:

- Alta concurrencia mediante un sistema denominado MVCC (Acceso concurrente multi-versión) que permite que mientras un proceso escribe en una tabla, otro proceso podrá acceder a la misma tabla sin necesidad de bloqueos. Esta característica es superior a los bloqueos de tabla o fila que aplican otros sistemas de gestión de base de datos.

- Debido a que es un proyecto Open Source y que hay una gran cantidad de desarrolladores detrás de este, existen muchos plugins para instalar como el que se utilizara en el proyecto para manejar datos espaciales.

Quizás la gran ventaja de que Gsiam maneje los datos de los PDI y usar PostgreSQL para esto, es que se tiene absoluto control y no se depende de ningún servicio externo que podría ocasionar problemas de disponibilidad del servicio. Como por ejemplo en el

caso de Simplegeo se corre el riesgo de caer en gastos que no son admitidos en este proyecto ya que se trata de que sea un servicio completamente gratis para el usuario.

9.3.3.3 Desventajas

La desventaja más importante que se encontró en manejar los PDI es la inexperiencia del equipo en este tipo de tecnología. Esto genera que se tenga que invertir una cantidad de tiempo considerable en el aprendizaje de base de datos espaciales y consultas espaciales que se utilizaran en el desarrollo del proyecto.

9.3.4 Conclusiones

Por lo expuesto anteriormente se opto por manejar los PDI en una base de datos propia PostgreSQL con el plugin PostGis para el manejo de datos geográficos.
Con esta elección se tendrá más control sobre los datos manejados y no se dependerá de servicios externos.
Para el acceso a datos se usara JDBC ya que JPA, que permite una integración con EJB de manera clara, no soporta consultas espaciales.
Si bien se manejaran datos geográficos en la base de datos estos serán solo para almacenar la ubicación geográfica de los PDI ya que manejarlos sin esta característica resultaría por demás costoso en cuanto a rendimiento.

9.4 Creación de Servicios Web

9.4.1 Introducción

Dado que se decidió manipular los datos en una base de datos propia es necesario construir y exponer un conjunto servicios web para poder acceder a la información. Si hablamos de servicios web en Java, no se puede pasar por alto el API JAX-WS [*ref.029*] y [*ref.030*]. Esta API Java esta dentro de la especificación JEE 5 y permite la definición de servicios web mediante anotaciones de una forma simple. El intercambio de información con un cliente es a partir de mensajes XML como lo hace SOAP. JAX-WS no permite, de forma estándar el intercambio de información mediante el formato JSON [*ref.031*] que es más liviano que el XML.
Es de vital importancia lograr una comunicación rápida y eficiente entre el dispositivo móvil y el servidor. Las dos formas más utilizadas hoy en día para construir servicios web son: mediante la tecnología SOAP + WSDL o basados en la arquitectura REST. Es por esto que surge la interrogante acerca de la elección de la misma.

9.4.2 WSDL + SOAP

Una de las primeras alternativas que surge al momento de hablar de consumir servicios es la utilización de SOAP y WSDL que son dos de los estándares más usados de la especificación WS.
Por un lado SOAP [*ref.032*], es un estándar que permite invocar servicios remotos mediante XML. Otra de sus características es que es independiente de la arquitectura de red por cual se puede invocar servicios sobre cualquier protocolo como puede ser SMTP, MQ entre otros aunque en el proyecto el único que interesa es HTTP.
Por otro lado, WSDL [*ref.033*] es el estándar en XML que permite definir servicios como un contrato que el cliente debe saber para lograr la comunicación.

Las funcionalidades generales de estos dos estándares son la de consumir servicios sobre casi cualquier arquitectura. Esto en una arquitectura compleja y que requiera de una integración con sistemas legados podría ser la opción adecuada.

9.4.2.1 Ventajas

Quizás la gran ventaja que se encuentra al utilizar SOAP y WSDL es la madurez con la que cuentan estos estándares, documentación y productos que hay para su desarrollo.

9.4.2.2 Desventajas

Ya grandes proveedores de Web 2.0 están migrando a una arquitectura REST, incluyendo a Yahoo, Google, eBay y Facebook, quienes marcaron como obsoletos a sus servicios SOAP y WSDL y pasaron a usar un modelo más fácil de usar, orientado a los recursos.
Esto se debe quizás a la complejidad que se tiene para desarrollar y consumir servicios con SOAP y WSDL y al ancho de banda que consumen.

9.4.3 REST

En una arquitectura basada en REST todo es un recurso. Un recurso se accede a través de una interfaz común basada en los métodos estándar HTTP. En una arquitectura REST se suele tener un servidor REST que proporciona el acceso a los recursos y al cliente REST que accede y modifica los recursos REST. Todos los recursos deben soportar las operaciones HTTP comunes [*ref.034*].
Los recursos son identificados por un Global ID (por lo general son URIs).
REST permite que los recursos tengan diferentes representaciones, ej. texto, XML, JSON, etc.

El cliente REST puede pedir una representación específica a través del protocolo HTTP (Negociación de Contenido).

RESTFul: Son aquellos servicios Web que funcionan bajo REST [*ref.035*].
Los Web Service REST proporcionan acceso a través de los métodos GET y POST de HTTP.
GET: En accesos vía GET, tanto las operaciones como los parámetros se pasan por la URL. Sólo soporta argumentos con tipos simples. Es utilizado para obtener un recurso del servidor
POST: En este tipo de acceso, la información no viaja en mensajes SOAP Envelope, sino directamente en el payload del mensaje. Se utiliza para agregar un recurso en el servidor
PUT: Este método es usado para cambiar el estado de un recurso o actualizarlo
DELETE: Este método es utilizado para eliminar un recurso.
Una implementación concreta de un servicio web REST sigue cuatro principios de diseño fundamentales:

- Utiliza los métodos HTTP de manera explícita
- No mantiene estado
- Expone URIs con forma de directorios
- Transfiere XML, JavaScript Object Notation (JSON), o ambos

9.4.3.1 Ventajas

- Simples de implementar.
- Es bueno cuándo es importante que la comunicación sea ligera en términos de bytes transmitidos debido al coste.
- Eficientes cuándo el consumidor tiene limitaciones de ancho de banda ej: móviles, PDAs, etc.
- Ya varios grandes proveedores de Web 2.0 están migrando a esta tecnología, incluyendo a Yahoo, Google, eBay y Facebook, obteniendo un modelo más fácil de usar orientado a los recursos.

.

9.4.3.2 Desventajas

- Gran número de objetos.
- Manejar el espacio de nombres (URIs) puede ser engorroso.
- La descripción sintáctica/semántica muy informal (orientada al usuario).
- Pocas herramientas de desarrollo.

9.4.4 Conclusiones

Tomando en cuenta que la aplicación está enfocada en dispositivos móviles se determina que la mejor opción es utilizar REST dado que es ideal para este tipo de aplicaciones como se describió anteriormente. Además los servicios web en REST son muy fáciles de implementar y dado que la complejidad principal del proyecto es Android no existe la necesidad de sumar complicaciones técnicas.
RestEasy de Jboss es el framework elegido para la implementación de los servicios web RestFul. RestEasy es una implementación de la especificación Java de JAX-RS. Dicha especificación define una API Java sobre protocolo HTTP que permite definir servicios web RestFul. Se escogió este framework ya que además de ser Open Source, su integración con el servidor Jboss es muy sencilla aunque puede correr en cualquier contenedor de Servlets como podría ser Tomcat, lo cual lo dota de una gran portabilidad.
Ya que JAX-RS es una especificación del lado del servidor RestEasy permite definir su contraparte del lado del cliente y así implementar los métodos RestFul mediante anotaciones definidas en la especificación JAX-RS.

9.5 Interfaz Grafica

9.5.1 Introducción

Al momento de comenzar con los prototipos de pantallas el equipo encontró que la parte grafica no era sencilla, ya que cada componente se maneja de forma particular y cuentan con cierta complejidad.
Dentro de los prototipos de pantallas se tienen componentes del estilo Tabs, ActionBar, List, Maps, SegmentedHost, entre otros.
Es por esto que surge la necesidad de análisis de la interfaz grafica. Además de los componentes que vienen por defecto dentro del SDK Android se investiga una librería llamada GreenDroid [*ref.036*]. Esta librería intenta ayudar a los desarrolladores codificar aplicaciones altamente funcionales así como también aprovechar toda la potencia de Android. En el mercado se encuentran varias aplicaciones que utilizan esta librería.

9.5.2 Componentes por Defecto

Android proporciona componentes por defecto, llamados widgets o las herramientas necesarias para crearlos.
Para desarrollar la interfaz grafica, se puede utilizar ficheros xml con un editor grafico proporcionando por el framework, o desarrollar la interfaz en base a código Java.

9.5.2.1 Ventajas

Utilizar los componentes por defecto del framework tiene la gran ventaja de contar con el testing necesario para su óptima utilización. Además de contar con una cantidad de ejemplos y tutoriales para su aprendizaje.

9.5.2.2 Desventajas

Los componentes estándares tienen un aspecto básico y no son por demás atractivos para el usuario final. Esto es importante en una aplicación para dispositivos móviles, donde los usuarios buscan interfaces atractivas y sencillas.
Otra desventaja que presenta es que se repite mucho código para implementar interfaces sencillas.

9.5.3 GreenDroid

GreenDroid es una librería para Android, que facilita el desarrollo de la interfaz grafica además de hacerla mucho más amigable para el usuario. Uno de los propósitos de GreenDroid es evitar la pérdida de tiempo en la copia de los mismos fragmentos de código una y otra vez. Dentro de sus principales características, se destaca la inclusión de una barra superior (ActionBar) con diferentes funcionalidades configurables que le dan al usuario mayor usabilidad. Otra de las características es la facilidad de creación y manejo de pestañas (tabs) dentro de una aplicación. Para utilizarlo en un proyecto Android, solo basta con incluir el proyecto GreenDroid al workspace y luego incluirlo como librería al proyecto. A continuación se muestran algunos ejemplos.

[*ref.037*]: Ejemplos GreenDroid

9.5.3.1 Ventajas

Una de las grandes ventajas es que es fácil de utilizar. Al utilizar Greendroid se garantiza un diseño y estilo único en toda la aplicación. Es Open Source y como se menciono anteriormente es de gran importancia para el proyecto.

9.5.3.2 Desventajas

Por tratarse de una librería relativamente nueva no se encuentra mucha información, solo se encuentra la documentación del sitio y ejemplos en el Blog del autor.

9.5.4 Conclusiones

Por lo expuesto anteriormente se opto por el uso de GreenDroid para el proyecto.
Debido a que Gsiam apunta a ser una herramienta de uso cotidiano, es primordial que sea amigable. Además al ser un prototipo es de suma importancia que sea mantenible y reutilizable para poder ser modificado en el futuro de manera sencilla.
En los prototipos de pantallas se ven varios componentes los cuales al ser tratados con esta librería se logra un aspecto más profesional, entre ellos se encuentran los tabs, actionBar y list.
A continuación se muestra un prototipo de tab con los componentes por defecto y otro con GreenDroid para notar la diferencia visual:

10 Estudio de Factibilidad

10.1 Factibilidad Económica

No existe un factor de costo económico, ya que se trata de un proyecto de grado universitario, por ende, la mano de obra involucrada no es considerada en los costos del proyecto. A demás dicho proyecto será lanzado a la comunidad Open Source por lo que todas las herramientas utilizadas respetan esta filosofía.

10.2 Factibilidad Técnica

El estudio de factibilidad técnica se realiza mediante una prueba de concepto en una etapa temprana del proyecto. Considerando la definición del alcance del proyecto y el conocimiento técnico del equipo, se identifican tres funcionalidades que se presentan como determinantes para evaluar la factibilidad técnica:

- Llamadas a servicios externos mediante REST
- Visualizar un mapa de google maps con una posición determinada
- Conexión e integración con la red social Facebook.

Cabe destacar que los conocimientos técnicos del tutor son de gran aporte y sirven como guía a la hora de afrontar este tipo de problemas técnicos.

10.2.1 Tecnología Involucrada

10.2.1.1 SpringAndoid

Para la llamadas a servicios Web REST desde el dispositivo Android se utilizo un framework de Spring llamado SpringAndroid que facilita la llamada y parseo de los mensaje devueltos por el servicio.
Abajo se muestra una simple llamada a un servicio con una determinada URL que devuelve el resultado como String.

```
RestTemplate restTemplate = new RestTemplate(new
HttpComponentsClientHttpRequestFactory());
String result = restTemplate.getForObject(url, String.class);
```

10.2.1.2 RestEasy

Para realizar y publicar los servicios Rest del lado del servidor de aplicaciones se utilizo el framework RestEasy de Jboss que se integra fácilmente a Jboss Application Server que es el servidor elegido por el equipo.
A continuación se muestra como definir el servicio con annotation de una forma fácil e intuitiva.

```
@GET
@Path("/login/{email}/{pass}")
@Produces("application/json")
public UsuarioDTO login(@PathParam("email") String email, @PathParam("pass")
String pass)
```

Esta firma indica que este servicio acepta solicitudes GET, la URL en la cual estará publicado y que la salida de método será un usuario con formato JSON que es un formato ligero para el intercambio de información parecido al XML pero más liviano y fácil de leer. Cabe aclarar que la seguridad no fue un punto a considerar en el proyecto.

Desde el cliente móvil la llamada a este servicio y el parseo de la respuesta se realiza de una forma muy sencilla.

```
Map<String, String> parms = new HashMap<String, String>();
parms.put("email", email);
parms.put("pass", pass);
UsuarioDTO user = restTemp.getForObject(
Constantes.LOGIN_SERVICE_URL, UsuarioDTO.class, parms);
```

De esta forma se está llamando al servicio antes descrito pasándole el email y contraseña para el login en el sistema. El parámetro del UsuarioDTO.class está diciendo que el retorno será de este tipo y el parseo lo realizara SpringAndroid de forma transparente.

10.2.1.3 Facebook API

Facebook proporciona una librería Open Source, para la integración desde Android con los servicios que este publica [*ref.038*]. Dicha librería, permite entre otras cosas, publicar información, sea un texto o una imagen, en el muro de facebook de forma sencilla. Este es uno de los requerimientos que se plantearon como importantes en el proyecto, por lo tanto es de prioridad su realización.

Las siguientes líneas permiten la autentificación en facebook para realizar las llamadas a los distintos servicios que ofrece.
EL APP_ID es un código de autenticación proporcionado por facebook al registrar la aplicación.

```
FaceBook facebook = new Facebook(APP_ID);
facebook.authorize(this,getResources().getStringArray(R.array.permissions),
new FaceBookDialog());
```

El siguiente código, permite publicar un mensaje en el muro del usuario autenticado.

```
Bundle parameters = new Bundle();
parameters.putString("message", msg);
String response = facebook.request("me/feed", parameters, "POST");
```

También se puede publicar una foto de esta manera:

```
Bundle parameters = new Bundle();
parameters.putByteArray("picture", byte[] foto);
String response = facebook.request("me/photos", parameters, "POST");
```

A continuación se muestra un fragmento de código el cual permite obtener los amigos del usuario en Facebook. Estos son devueltos en formato JSON y luego son parseados para desplegarlos en el dispositivo móvil. Los datos devueltos son, el identificador del usuario en Facebook, el nombre y la foto de perfil.

```
String respuesta = InvitarAmigosActivity.facebook.request("me/friends", params);
jsonArray = new JSONObject(respuesta).getJSONArray("data");
AmigoFacebook amigo;
if (jsonArray.length() > 0) {
        for (int i = 0; i < jsonArray.length(); i++) {
                JSONObject jo = jsonArray.getJSONObject(i);
                amigo = new AmigoFacebook();
                amigo.setId(jo.getString("id"));
                amigo.setNombre(jo.getString("name"));
                amigo.setFotoUrl(jo.getString("picture"));
                amigo.setSeleccionado(false);
                list.add(amigo);
                }
        }
```

Además de las funcionalidades anteriores el equipo evaluó e investigo la posibilidad de implementar el login de la aplicación vía Facebook, es decir, utilizar el usuario de Facebook en vez de crear uno propio de Gsiam. Esto es muy utilizado actualmente por muchas aplicaciones, para que el usuario no tenga que perder tiempo al crear una nueva cuenta en otro sistema y tener que recordar varias contraseñas.
El equipo prefirió en esta etapa del proyecto tener más control sobre la aplicación y no depender de ningún servicio externo que pudiera dejar el sistema fuera de servicio. Sin embargo, se entiende que esta es una funcionalidad importante y se agregará a los trabajos futuros.

10.3Conclusión

Como se menciono anteriormente, la factibilidad económica no fue considerada por tratarse de un proyecto académico y Open Source.
Con respecto a la factibilidad técnica, las tres problemáticas planteadas fueron resueltas de manera exitosa recogiendo los siguientes resultados:

- La comunicación entre, el servidor de aplicaciones donde serán publicados los servicios webs y la aplicación Android corriendo en un dispositivo móvil.
- Visualización de un mapa con una marca en un punto geográfico determinado.
- Publicación de mensajes e imágenes en Facebook así como también el acceso a la lista de amigos de esta red.

En base a estos resultados, el equipo cree que técnicamente es factible la realización del proyecto. Se puede concluir que los conocimientos adquiridos en esta etapa del proyecto serán de vital importancia para la correcta ejecución del mismo, así como también el código generado para esta prueba servirá como base para el comienzo de la implementación de Gsiam.

Capítulo IV. Implementación

11 Especificación de Requerimientos de Software ERS

11.1Introducción

11.1.1 Propósito

El ERS formaliza los requerimientos para el proyecto “Geolocalización de Sitios de Interés Para Teléfonos Móviles”, en el contexto de Proyecto de Grado de la carrera Licenciatura en Informática. Debido a la complejidad en que se da el proyecto, este documento especifica los requerimientos del producto de software de forma clara y completa para que el sistema funcione adecuadamente.

El ERS que utiliza el equipo de proyecto, se basa en el resumen de la norma IEEE Std 830 – 1998 [*ref.039*], visto en la asignatura “Ingeniería de Software II” en el tercer año de la carrea de “Licenciatura en Informática” y material del curso y documento de Anteproyecto generado en la pre-etapa de proyecto.

11.1.2 Audiencia

El ERS está dirigido a toda persona que desee tomar contacto con el proyecto, y en particular a los actores que desarrollan y evalúan el mismo, principalmente será utilizado por los diseñadores del sistema y los que puedan requerir el mantenimiento del mismo. Permitirá entender en forma detallada todos los aspectos del sistema, además será indispensable a la hora de realizar las pruebas.

11.1.3 Alcance

El alcance del proyecto es construir una aplicación Java para dispositivos móviles “G-SIAM” basada en la Geolocalización de Sitios de Interés, esto es, principalmente proveer información de todos los sitios de interés ubicados a una distancia dada de un dispositivo móvil en tiempo real.

Además G-SIAM contara con el soporte para la administración de usuarios y contactos (amigos), así como también se integrara con otras redes sociales para poder interactuar con las mismas ya sea al publicar información acerca de los sitios de interés o invitando a nuevos usuarios.
El sistema se desarrollara bajo la plataforma Android para dispositivos móviles. El mismo está bajo la licencia Open Source.

La principal meta de este proyecto es ver las tecnologías que están en vanguardia y mostrar lo que se pueden hacer con las mismas. El sistema que se concebirá en el marco del proyecto de grado no es más que un prototipo con el cual en un futuro se podría convertir en un producto maduro con el nivel necesario para poder competir con los estándares del mercado.

11.1.4 Limitaciones al Alcance

Los siguientes puntos fueron excluidos del alcance del proyecto dado los costos que estos implican, no obstante los mismos podrían incluirse en una futura versión:

- Cabe destacar que aunque el sistema corra sobre la plataforma Android, esto no significa que correrá en cualquier celular, al menos lo hará en un determinado modelo.
- No se realizarán tareas de optimización del servidor/motor de base de datos.
- Traducción de los mensajes del sistema a los distintos idiomas que éste soportará.

11.2Descripción General

11.2.1 Perspectiva del producto

Al concebir G-SIAM se decide crear un sistema que aproveche al máximo las tecnologías de vanguardia, al estar en pleno crecimiento tecnológico existen cada vez mas dispositivos móviles con acceso a Internet y GPS integrado, así como también mas conectividad mediante redes sociales.
Si bien el problema que se ataca en particular es la implementación de una aplicación Android para dispositivos móviles, la preocupación de fondo es cumplir con la actual demanda del mercado, esto consiste en brindar acceso a la información de manera clara e inmediata y satisfacer la necesidad de vincularse mediante redes sociales para comunicarse con personas con intereses en común.

El producto va de la mano al desarrollo actual de la tecnología y aporta tanto a la parte tecnológica como a la sociedad en sí, ya que pretende ser una herramienta de uso popular enfocada a la comunicación y el entretenimiento.

11.2.2 Funciones del producto

Sitios de Interés

- Localización geográfica de sitios de interés.
- Visualizar en un mapa los sitios de interés.
- Filtrar las búsquedas de los sitios de interés.
- Identificación del sitio más cercano.
- Compartir sitios de interés.
- Creación, modificación y eliminación de sitios de interés.

Publicaciones

- Publicar comentarios o fotos a cerca del sitio de interés.
- Visualización de las publicaciones de cada lugar de interés.
- Rankear (darle un puntaje) a los sitios de interés.

Usuarios y Contactos

- Creación, modificación y eliminación de cuentas de usuario para el uso de la aplicación.
- Cada usuario podrá formar su red de contactos/amigos.
- Compartir la ubicación del usuario con la lista de sus contactos.
- Ver contactos que estén cerca de la ubicación del usuario.

Conectividad Externa

- Publicar comentarios y fotos acerca de los sitios de interés en Facebook.
- Agregar contactos del usuario vía Facebook o mail.

11.2.3 Características del usuario

Los usuarios de G-SIAM son personas sin ningún requisito adicional para poder utilizar el sistema.
Para usuarios con experiencia en uso de sistemas de Geolocalización, G-SIAM significa la posibilidad de tener varias herramientas en una sola.
Para los usuarios sin experiencia en uso de estos sistemas, G-SIAM podría ser una buena puerta de entrada al mundo de la Geolocalización dado su sencillez y amigabilidad. Además de ser una oportunidad de relacionarse con este tipo de aplicaciones ya que mercado y el futuro apuntan hacia las mismas.

Para personas en calidad de turistas se presenta como una buena herramienta de viaje, ya que el producto le brinda información en tiempo real acerca de los lugares que están a su alrededor.
Para usuarios comerciantes es una buena oportunidad para publicitar sus negocios.

11.2.4 Restricciones

El Sistema Operativo en el que corra la aplicación móvil debe ser Android.
El dispositivo móvil debe contar con conexión a Internet

11.3Requisitos Específicos

11.3.1 Requerimientos Funcionales

Sitios de Interés

- Localización geográfica de sitios de interés.

G-SIAM deberá permitir la localización geográfica de lugares de interés (Farmacias, restaurantes, pubs, hoteles, etc.) en un radio configurable a la ubicación del móvil. Deberán mostrarse los diferentes lugares identificados con iconos dependiendo del lugar.

- Visualizar en un mapa los sitios de interés.

GSIAM permitirá visualizar en un mapa los sitios de interés que retorne la búsqueda así como también la posición actual del usuario. El mapa deberá mostrar los nombres de las calles cercanas para su mayor entendimiento y ubicación.

- Filtrar las búsquedas de los sitios de interés.

Se debe poder aplicar filtros para la búsqueda de un sitio, como por ejemplo, por la categoría del sitio o por ranking.

- Identificación del sitio más cercano.

El sistema deberá saber fácilmente cual de todos los sitios de interés de la búsqueda es el más cercano a la ubicación del móvil. Los mismos estarán en orden de proximidad.

- Compartir sitios de interés.

El sistema tendrá la opción para que el usuario pueda recomendar el sitio vía mail.

- Creación, modificación y eliminación de sitios de interés.

El sistema debe brindar la posibilidad de agregar, modificar o eliminar un lugar de interés. Con esto se permite una mayor cantidad de sitios y una retroalimentación continua.

Publicaciones

- Publicar comentarios o fotos a cerca del sitio de interés.

Se podrán subir comentarios a cerca del lugar de interés, así como también fotos.

- Visualización de las publicaciones de cada sitio de interés.

Al seleccionar algún sitio de interés, el mismo debe tener la opción para mostrar todas las publicaciones subidas por los demás usuarios.

- Rankear (darle un puntaje) a los sitios de interés.

El sistema permitirá al usuario rankear, es decir, darle un puntaje del 1 al 5 al sitio que selecciono según su criterio de aceptación (1 pésimo lugar, 5 excelente lugar).

Usuarios y Contactos

- Creación, modificación y eliminación de cuentas de usuario para el uso de la aplicación.

El sistema deberá tener un ABM de usuarios, de esta manera se podrá controlar el acceso a los usuarios mediante un log in.

- Cada usuario podrá formar su red de contactos/amigos

Vinculación entre usuarios mediante lista de Contactos, cada usuario podrá formar su red de contactos/amigos para poder acceder a ciertas funcionalidades de la aplicación.

- Compartir la ubicación del usuario con la lista de sus contactos.

Compartir la ubicación del usuario con la lista de sus contactos.

- Ver contactos que estén cerca de la ubicación del usuario.

G-SIAM será capaz de mostrar los contactos que se encuentran cerca de la ubicación del usuario.

Conectividad Externa

- Publicar comentarios o fotos acerca de los sitios de interés en Facebook.

El usuario tendrá la opción de mostrar sus publicaciones en redes sociales como Facebook y/o Twitter.

- Agregar contactos del usuario vía Facebook o mail.

El usuario podrá invitar/agregar a sus amigos al sistema, ya sea desde su cuenta Facebook o vía mail.

11.3.2 Requerimientos de Interfaz Externa

Para conectarse con los servicios expuestos por Facebook se exponen servicios mediante RESTFul:

https://graph.facebook.com

11.3.3 Restricciones de Diseño

El equipo de proyecto cuenta con la limitación de hardware para probar la aplicación en un dispositivo móvil. Por tal motivo solo se garantizara el correcto funcionamiento de la aplicación en un terminal el cual es propiedad del equipo. El modelo de dicho Smartphone es Samsung Galaxy S. A demás de probar la aplicación sobre este dispositivo, por motivos de practicidad se utilizara el emulador provisto por el sdk de Android para desarrollar la aplicación.
Cabe destacar que las versiones de Android soportadas serán la 2.1 hasta la 2.3.4

11.3.4 Atributos

Amigable: La idea del prototipo es que sea una herramienta enfocada a la comunicación y entretenimiento del usuario, por lo tanto la misma debe ser muy amigable y fácil de usar.

El sistema debe tener una interfaz de usuario apropiada y una documentación adecuada.

Mantenible: Principalmente enfocado en el mantenimiento adaptativo y perfectivo, pensando en la evolución del prototipo. Esta característica es crítica, ya que en el marco del proyecto se desarrolla un prototipo, el cual tiene que ser continuado en el futuro para construir un producto más competitivo en el mercado. No tiene que ser engorroso agregar los futuros requerimientos además de adaptarse a las necesidades de posibles cambios.

Confiable: El sistema va a estar en plena interacción con cuentas de redes sociales, por lo tanto el mismo debe ser muy confiable para lograr un buen producto que proteja al usuario.

Eficiente: Dado que el sistema va a correr en teléfonos celulares los mismos tienen limitaciones en consumo de ancho de banda así como también una capacidad de procesamiento menor a la de los sistemas convencionales, por esto mismo es necesario optimizar los tiempos de respuestas y hacer un uso económico de la memoria del dispositivo.

Verificable: Para lograr esto, es necesario contar con un buen análisis, una arquitectura bien definida, un correcto diseño y respetarlos estándares de programación.

Interoperable: Es importante la interoperabilidad del producto, ya que en este proyecto debe Interactuar con entidades externas (Facebook), y en el futuro pueda evolucionar e interactuar con más entidades**.**

11.4Otros Requerimientos

11.4.1 Tipo de aplicación

- Aplicación Android desarrollada en Java junto con el API “Spring Android” para acceder a servicios webs RESTful desde un dispositivo Android.
- Aplicación WEB basada en la especificación JEE a desplegar en un servidor web Jboss, para la misma se utilizara RestEasy así como también PostGis.

11.4.2 Invocación de la Aplicación

La aplicación será invocada manualmente a través de una interfaz Android.

11.4.3 Utilización de CPU y memoria

Hay que evitar un consumo excesivo de tiempo de procesamiento de CPU y aprovechar la utilización de memoria. Para ello es necesario optimizar los tiempos de respuestas.

11.4.4 Idioma y Formatos de Fecha

El idioma será español.

El formato de la fecha será ddmmyyyy hhmisss, siendo:

- yyyy = año
- mm = mes
- dd = día
- hh = hora
- mi = minutos
- ss = segundos

11.4.5 Logueo de errores y mensajes

Generación de un archivo de LOG con 3 niveles de detalle: DEBUG, ERROR e INFO.
La generación del archivo de LOG se realizará con la herramienta Log4J, que es una herramienta Open Source.

11.4.6 Control de Acceso a Usuarios y Seguridad

Este sistema no tiene requerimientos de seguridad específicos, al menos dentro del marco del proyecto de grado utilizaremos usuario y password. Para un futuro se podría utilizar el algoritmo de hash SHA para encriptar el password del usuario en la base de datos o se podría implementar un LDAP.

11.5Requerimientos Futuros

El contenido de esta sección surge luego de realizar una selección sobre el documento del Anteproyecto donde se determina cuales requisitos están dentro del alcance del proyecto y cuales se postergan para una etapa posterior al proyecto de grado.

Como se ve, la cantidad de productos orientados a la Geolocalización es infinita, el publico podría ser todo el mundo, se podría pensar en armar una base plataforma del producto y después construir varios productos dependiendo de la demanda, ya que podría enfocarse cada producto con la misma base pero orientado a distintos públicos.

11.5.1 Funciones del Producto

Sitios de Interés

- Estimación del tiempo de llegada a un sitio de interés.

Publicaciones

- Agregar más niveles de puntuación a la hora de Rankear.
- Sistema de puntos: sumar puntos al comentar, subir fotos, al puntuar sitios, etc.

Usuarios y Contactos

- Organización de eventos.
- Acceso a sitios visitados y comentarios de contactos.
- Visualización de información adicional de los contactos del usuario
- Chat para comunicarse con los contactos.

Conectividad Externa

- Interfaz para comercios para la gestión de promociones u ofertas.
- Reportes, estadísticas y gráficos para los comercios.
- Posibilidad de solicitar taxis desde su ubicación a través de la aplicación.
- Interfaz con las compañías de ómnibus.
- Integraciones con comercios para gestión de reservas, compras, etc.
- Visualización de información demográfica.
- Obtención del pronóstico del tiempo.
- Conexión con otras redes sociales.

11.5.2 Requerimientos Específicos

Sitios de Interés

- Estimación del tiempo de llegada a un sitio de interés.

Al seleccionar un sitio el usuario podrá obtener una estimación del tiempo que demoraría en llegar al mismo. (Caminando, en auto o en bus).

Publicaciones

- Agregar más niveles de puntuación a la hora de Rankear.

Se podría evaluar limpieza del lugar, servicio brindado, precio, etc.

- Sistema de puntos: sumar puntos al comentar, subir fotos, al puntuar sitios, etc.

Agregarle un sentido de juego a la aplicación, al tener esta opción podría ser más "vicioso" el uso del sistema incrementando la información de la red.

Usuarios y Contactos

- Organización de eventos.

Opción para poder organizar eventos, reuniones, etc. con los contactos del usuario.

- Acceso a sitios visitados y comentarios de contactos.

Al seleccionar detalles de un contacto, se podrá desplegar el historial de los sitios visitados y publicaciones del usuario.

- Visualización de información adicional de los contactos del usuario

Cada usuario tendrá un grupo de contactos, de los cuales tendrá privilegios, por ejemplo, la posibilidad de saber qué es lo que más le gusta, el lugar que comento más, o cual fue el último lugar donde se encontró.

- Chat para comunicarse con los contactos.

Al momento de ver los contactos cercanos, se podría activar un chat para poder interactuar con los contactos en línea.

Conectividad Externa

- Interfaz para comercios para la gestión de promociones u ofertas.

Brindarle a los comercios la opción de ingresar promociones u ofertas, así el usuario podría asociarse a los boletines de los sitios con lo cual si el mismo está en un radio determinado se le enviara la promoción u oferta a su celular.

- Reportes, estadísticas y gráficos para los comercios.

Relacionado al requerimiento anterior, se podría brindar información al comercio para poder administrar de manera más eficiente su negocio.

- Posibilidad de solicitar taxis desde su ubicación a través de la aplicación.

Construir Interfaces para las distintas compañías de servicios de taxis y de esta manera lograr solicitar un taxi desde un punto cualquiera dentro de una ciudad.

- Interfaz con las compañías de ómnibus.

Construir Interfaces para las distintas compañías de servicios de ómnibus y de esta manera lograr chequear en cuanto tiempo solicitar un taxi desde un punto cualquiera dentro de una ciudad.

- Integraciones con comercios para gestión de reservas, compras, etc.

Se pueden reservar lugares en restaurants, hoteles, comprar entradas a eventos, etc

- Visualización de información.

El usuario podrá saber información de la ciudad en la que se encuentra actualmente como por ejemplo, población, historia, índices, etc.

- Obtención del pronóstico del tiempo.

El usuario podrá saber el clima actual de la ciudad en la que se encuentra actualmente así como también el pronóstico de los próximos días.

- Conexión con otras redes sociales.

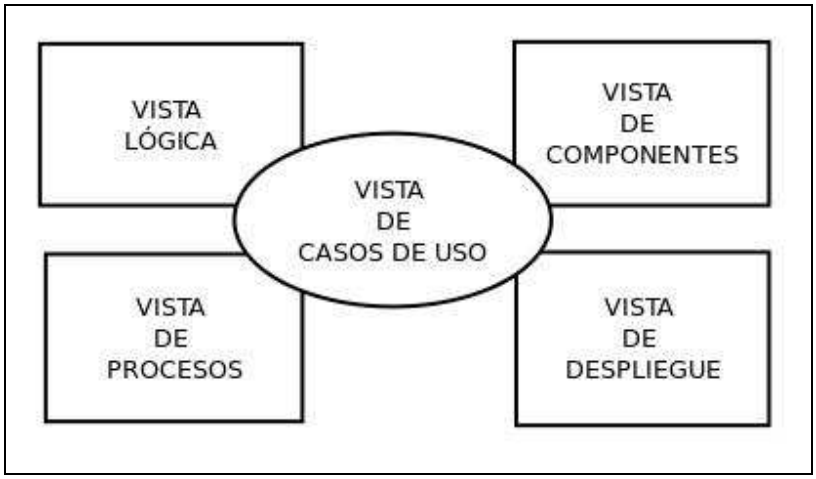
Las mismas integraciones con Facebook se podrían hacer con otras redes (Twitter, Fotolog, Myspace, etc).

12 Arquitectura de la Solución

12.1Introducción

Para documentar esta arquitectura se seguirá el modelo arquitectónico 4+1 que define las siguientes vistas.

- Vista Lógica
- Vista de Procesos
- Vista de Implementación
- Vista de Despliegue
- Escenarios de caso de uso

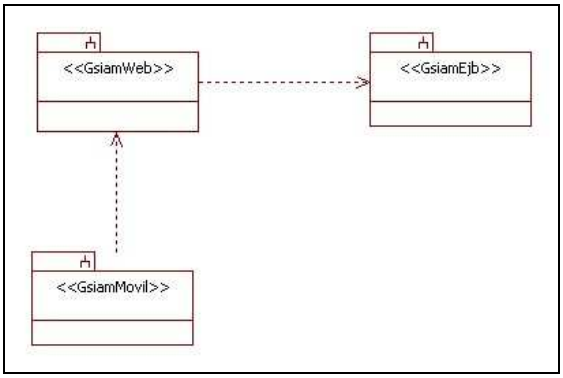


Dicho modelo es una forma fácil y apropiada de definir la arquitectura de un sistema ya que nos permite tener diferentes enfoques del sistema.

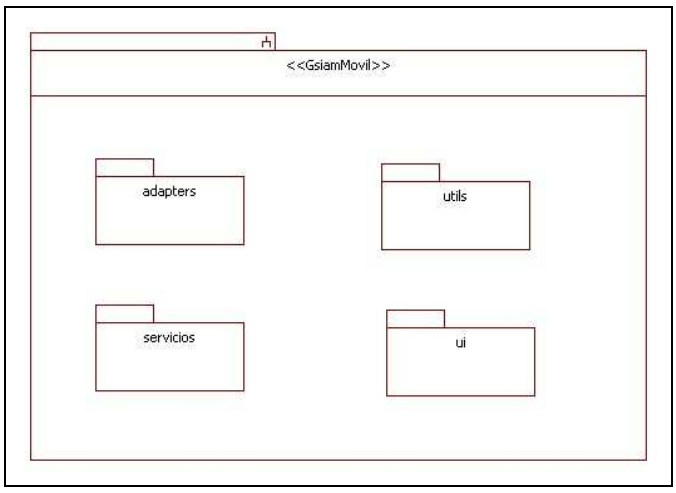
12.2Vista Lógica

La vista lógica representa básicamente los requerimientos funcionales del sistema. El sistema se descompone en objetos que nos permiten determinar patrones de comunes en el sistema que se podrían reutilizar.

El equipo dividió la vista lógica en subsistemas para luego estudiar de forma más concreta cada uno. Esto nos permite tener una visión general del sistema para después si estudiar cada uno de forma concreta.

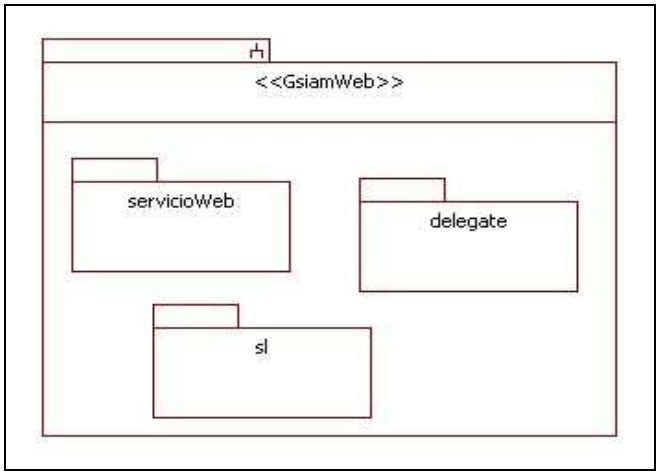


El diagrama muestra los tres subsistemas que componen el sistema y sus dependencias. El subsistema GsiamMovil es la aplicación para el sistema operativo Android que será el cliente del sistema. Dicho cliente consumirá los servicios publicado por el subsistema GsiamWeb. GsiamEjb es el encargado de realizar la lógica de negocio del sistema.



12.2.1 Catalogo de elementos

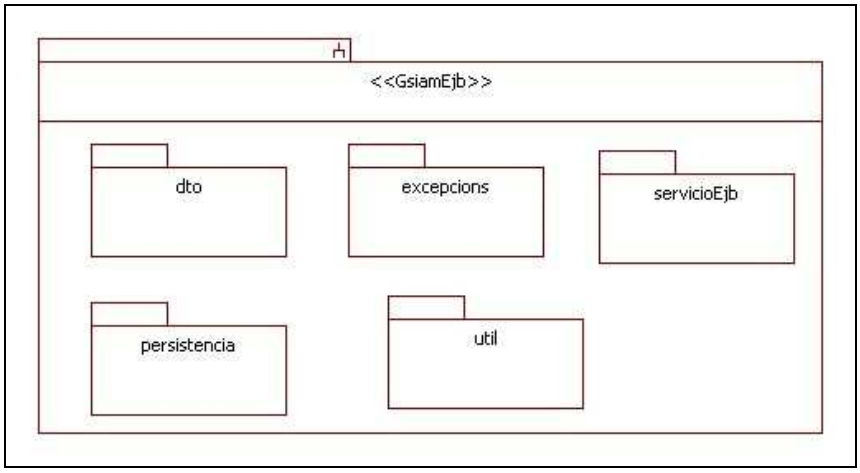
Paquete	Descripción
adapters	El paquete adapters es el encargado de definir los adaptadores para la interfaz del móvil. Estos objetos definen como se mostraran los datos por pantalla y son el puente entre los objetos y los datos propiamente dichos.
servicios	El paquete servicios es el que define los IntentServices de Android que consumirán servicios Restful publicados en la aplicación web. Los intentServices son una subclass de la clase Service de Android que nos permiten hacer peticiones de forma asincrónica mediante la implementación de un hilo de trabajo independiente del hilo principal.
ui	Este paquete contiene todas las pantallas de sistema. Representan la interfaz gráfica del móvil y en Android estas deben extender de la clase Activity
Utils	Este paquete contiene clases de utilidades que se utilizan en todo el sistema



12.2.2 Catalogo de elementos

Paquete	Descripción

servicioWeb	Este paquete contiene todos los servicios web que serán llamados desde la aplicación móvil. Se utiliza el modelo REST para implementar los servicios dado que es más sencillo y simple que SOAP + WSDL
delegate	El paquete delegate es el que se encarga de realizar la llamada al los EJB que ejecutaran la lógica de negocio.
sl	Este paquete contiene únicamente la clase ServiceLocator que básicamente lo que hace es localizar los EJB por su nombre JNDI



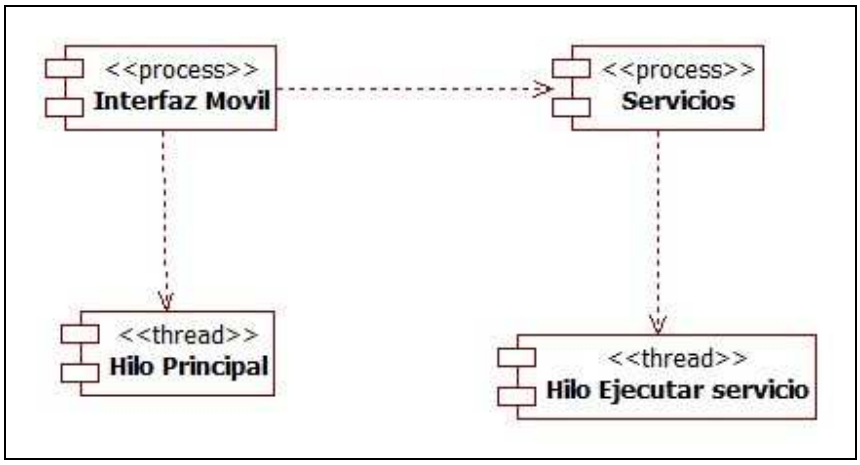
12.2.3 Catalogo de elementos

Paquete	Descripción
dto	El paquete dto contiene los objetos que viajaran entre las capas para así no pasar objetos de negocio sino una representación de estos.
excepcions	Este paquete contiene como su nombre lo indica las excepciones que serán lanzadas desde la lógica de negocio y se atraparan para en el proyecto web para transformarlas y presentárselas al usuario
servicioEjb	Este paquete contiene los EJB que implementaran la lógica de negocio del sistema.
persistencia	El paquete persistencia contiene las clases que persistirán los datos en la base de datos Postgres. Se implementa el patrón AbstractFactory y DAO

	para abstraernos del motor de persistencia.
utils	Este paquete contiene clases utilitarias que se utilizaran en todo el sistema.

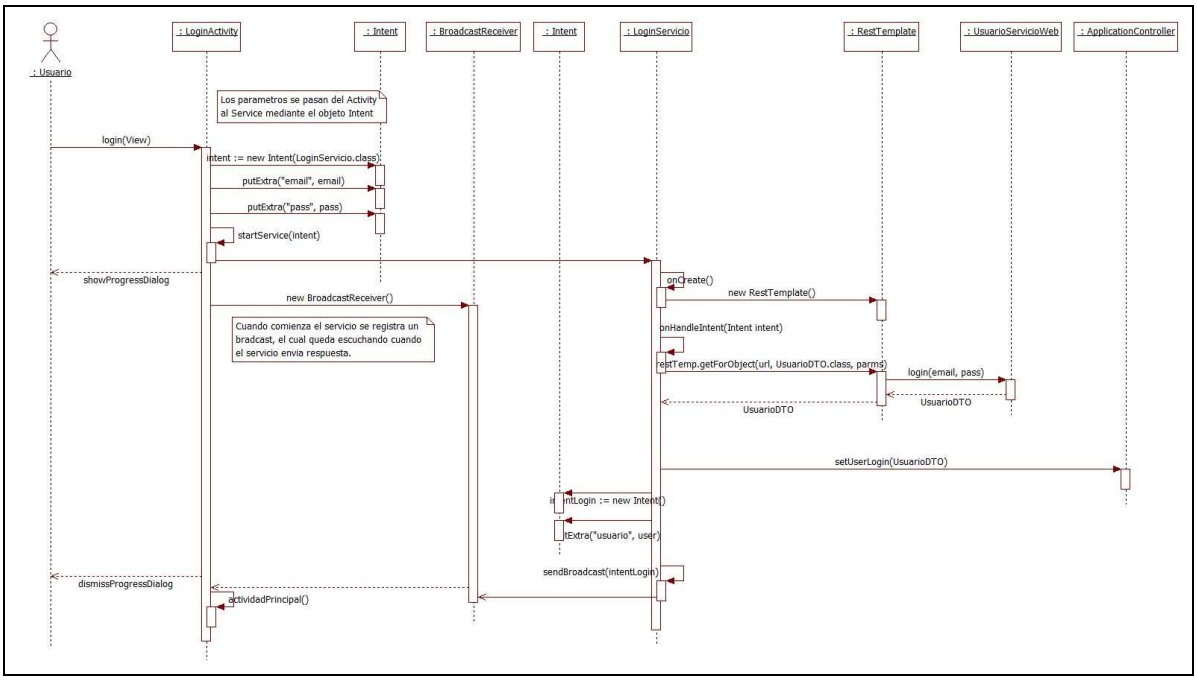
Para ver el diagrama de clases completo referirse al punto 7 de los anexos.

12.3Vista de procesos



Esta vista general muestra como interaccionan los módulos en el sistema. En particular se muestran como se manejan los hilos en el proyecto GsiamMovil. El proceso de interfaz móvil se ejecuta en un hilo que es el denominado hilo principal de la aplicación. El proceso de interfaz móvil invoca al proceso de servicios y este en un hilo independiente al principal ejecuta los servicios webs correspondientes. de esta manera no se interfiere con lo que el usuario está viendo.

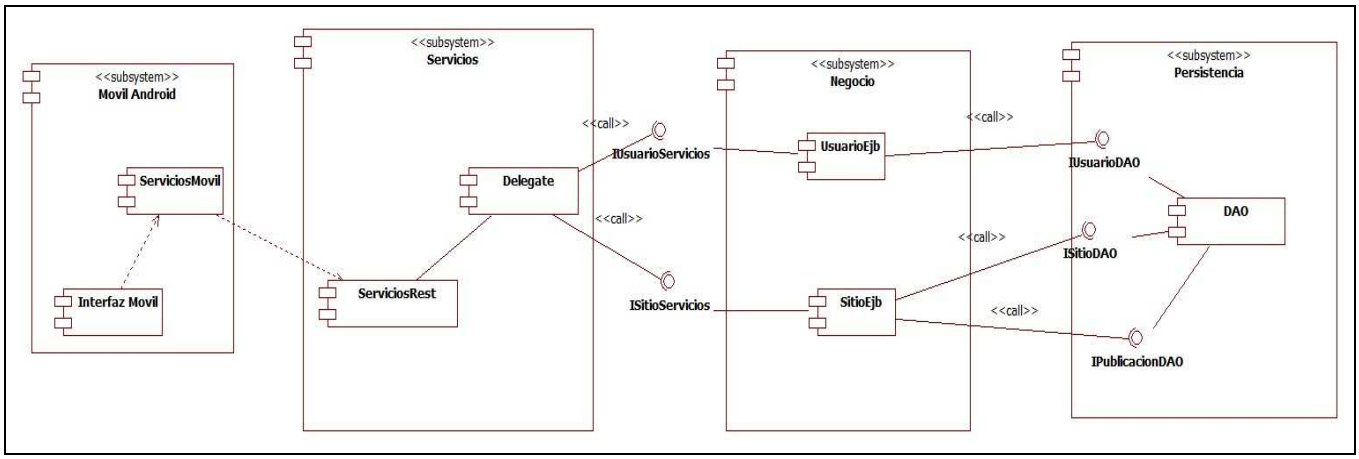
A continuación se muestra un diagrama de secuencia que detalle el proceso de login del sistema de forma detallada:



Para ver los demás diagramas de secuencia, referirse al punto 8 de los anexos.

12.4Vista de implementación

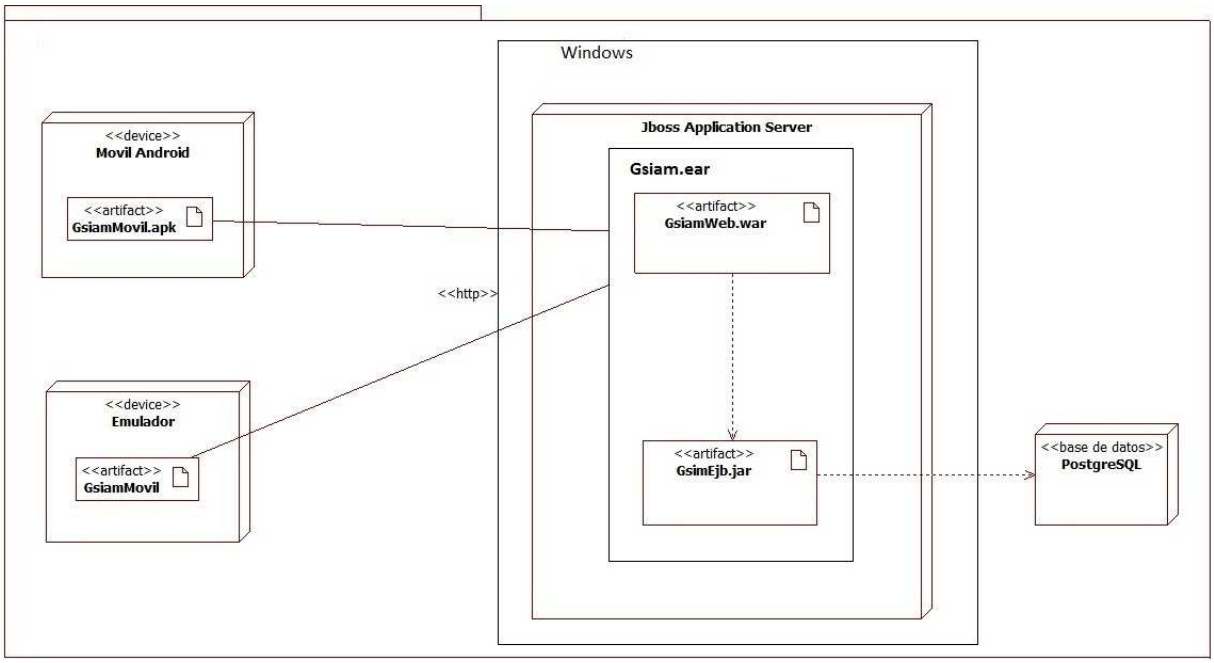
En esta vista se muestran los distintos componentes que conforman el sistema y como se distribuyen entre las diferentes capas.
El sistema se divide en cuatro capas bien marcadas como se muestra en el diagrama. Una capa del dispositivo móvil, otra con los Servicios Web Restful, la capa de negocio y la persistencia.



Componente	Descripción
Interfaz Móvil	Este componente es el encargado de capturar las acciones del usuario para luego procesarlas y ejecutar la lógica correspondiente.
Servicios Móvil	Este modulo contiene las invocaciones de los servicio Rest desde la aplicación móvil
ServiciosRest	Dicho componente representa a los servicios web publicados del lado del servidor de aplicaciones. Estos servicios serán invocados desde la aplicación instalada en el dispositivo móvil permitiendo la interacción entre los subsistemas.
UsuarioEjb	El componente UsuarioEjb representa al EJB encargado de procesar toda la lógica de negocio relacionada a los usuarios del sistema. IUsuarioServicio es la interfaz del EJB que será invocada desde la capa de servicios
SitioEjb	El EJB SitioEjb es el responsable de realizar toda la lógica de negocio referente a los sitios en el sistema. ISitioServicio es la interfaz del EJB que será invocada desde la capa de servicios
DAO	Este componente permite la interacción con la base de datos del sistema y la abstracción de como se guardan los datos. Se exponen tres interfaces IUsuarioDAO, ISitioDAO e IPublicacionDAO para la consulta o actualización de dichas entidades en la base de datos.

12.5Vista de Despliegue

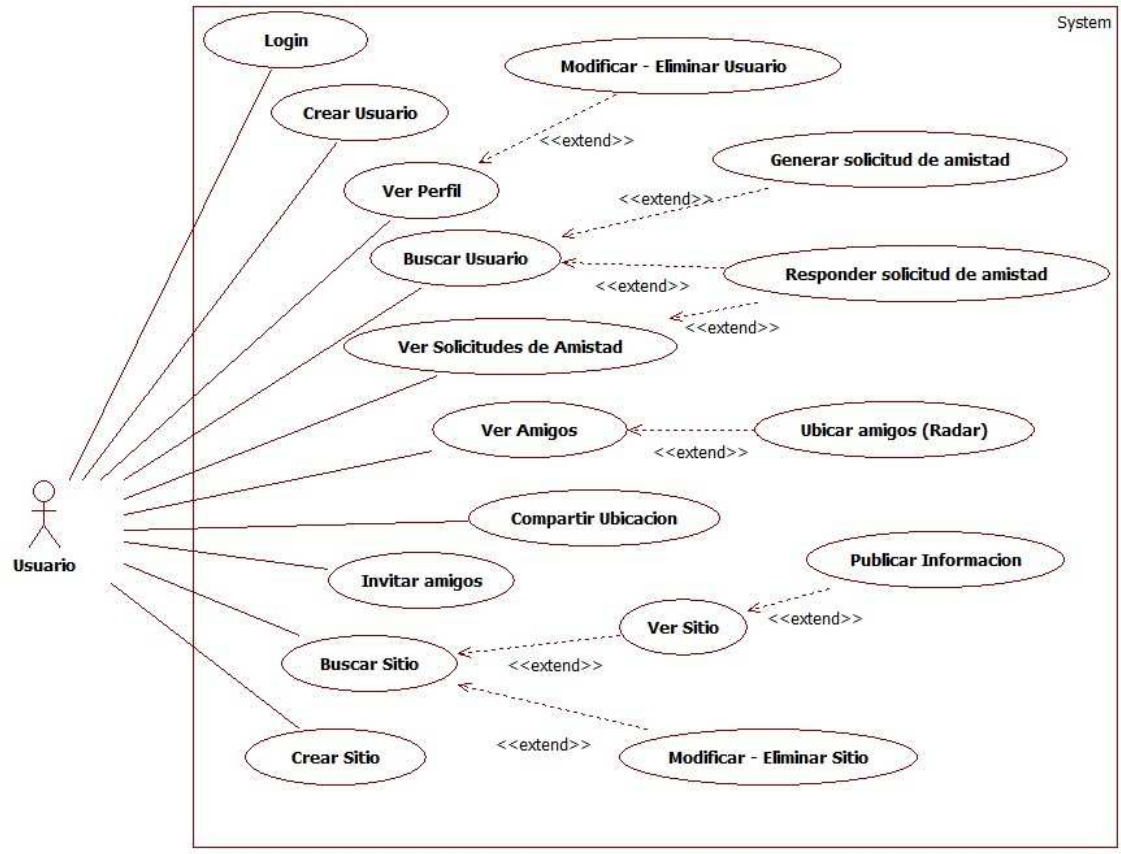
Esta vista del sistema nos permite determinar cómo se distribuyen las diferentes piezas de software en los diferentes componentes físicos.



Como servidor de aplicaciones se utilizara Jboss Application Server 6 en donde correrá el componente gsiam.ear compuesto por GsiamEjb.jar y GsiamWeb.war. Como motor de base de datos se usara PostgreSQL con el plugin PostGis para el manejo de datos espaciales.
El componente GsiamMovil.apk correrá en el emulador del dispositivo móvil o el dispositivo concreto y será el que contendrá las clases

12.6Escenarios de Casos de uso

Esta vista nos permitirá ver los diferentes requisitos funcionales del sistema a continuación se muestra el diagrama de casos de uso del sistema:



A continuación se muestra el caso de uso de login como ejemplo. Para ver todos los casos de uso referirse al punto 5.3 de los anexos.

CU-01	
Información del Caso de Uso	
Nombre:	Login
Descripción:	Este CU indica los pasos a seguir para poder ingresar a la aplicación.
Actores:	Usuario Móvil
Prioridad:	Alta
Puntos de Extensión:	N/A
Puntos de Inclusión	N/A
Sistemas Externos:	N/A
Precondiciones:	N/A
Resultado Esperado:	• Usuario autentificado en la aplicación. • Despliegue de página principal de la aplicación.
Referencias:	N/A

Descripción de los Flujos

Flujo Principal

Descripción de Alto Nivel

Los usuarios deben autentificarse al ingresar al sistema, el mismo les pedirá su e-mail y una contraseña, si es correcta la contraseña se permitirá el acceso, de lo contrario se lanzara un error.

Descripción Detallada (Paso a Paso)

1. Sistema despliega el formulario de Login:
   - E-mail
   - Contraseña
2. Usuario ingresa E-mail y Contraseña y oprime el botón “Entrar”.
3. Sistema obtiene los datos del formulario.
4. Si los datos obligatorios son vacíos [A1]
5. Si el e-mail ingresado tiene un formato invalido [A2]
6. Si los datos no se encuentran registrados en el sistema [A3]
7. Sistema deja al usuario autentificado y despliega pantalla principal del sistema mostrando su nombre.
8. Fin caso de uso.

Flujos Alternativos

A1 - Datos obligatorios vacíos	[ ] Excepción [X] Validación [ ] Otro	**Descripción de Alto Nivel** Ninguno de los campos obligatorios deben ser vacios, por lo tanto se le envía un mensaje de error al usuario. **Descripción Detallada (Paso a Paso)** 1. El sistema envía un mensaje de error al usuario indicando que cierto campo no puede ser vacio. 2. El flujo vuelve al paso 2 del Principal.
A2 - Formato de e-mail invalido	[ ] Excepción [X] Validación [ ] Otro	**Descripción de Alto Nivel** La dirección de e-mail ingresada por el usuario debe ser correcta (Utilizando expresiones regulares) **Descripción Detallada (Paso a Paso)** 1. El sistema envía un mensaje de error al usuario indicando que esa dirección no es correcta. 2. El flujo vuelve al paso 2 del Principal

A3 - Error de Login	[X] Excepción [ ] Validación [ ] Otro	Descripción de Alto Nivel
		El e-mail ingresado debe ser válido y existir en el sistema y la contraseña debe pertenecer al mismo, por lo tanto se le envía un mensaje de error al usuario.
		Descripción Detallada (Paso a Paso)
		1. El sistema envía un mensaje de error al usuario indicando que el e-mail no existe, esta inválido o que la contraseña es incorrecta. 2. El flujo vuelve al paso 2 del Principal.

Estructura de Datos				
Campo	**Descripción**	**Tipo**	**Tamaño**	**Obligatorio**
E-mail	E-mail del usuario con el cual se identificara en el sistema	Texto	30	Si
Contraseña	Contraseña del usuario	Texto Oculto	20	Si

Capítulo V. Trabajos Futuros

1 Trabajos Futuros

Dados los buenos resultados obtenidos sin previa experiencia en las tecnologías involucradas y el plazo acotado que se tuvo para realizar el proyecto, es factible continuar investigando sobre esta línea para desarrollar requerimientos que no se pudieron incorporar en esta liberación.
Gsiam puede ser utilizado como base de posibles proyectos futuros de Geolocalización sobre dispositivos móviles y que tengan como premisa la ideología Open Source.
Cabe destacar que la aplicación desarrollada es un prototipo que aunque incorpora muchas funcionalidades interesantes necesita de muchas mejoras y pruebas para ser un producto final.
Muchos fueron los requerimientos que por falta de tiempo o por priorizar otros no se pudieron desarrollar. A continuación se detallan los mismos:

Sitios de Interés
- Estimación del tiempo de llegada a un sitio de interés.

Publicaciones
- Agregar más niveles de puntuación a la hora de Rankear.
- Sistema de puntos: sumar puntos al comentar, subir fotos, al puntuar sitios, etc.

Usuarios y Contactos
- Organización de eventos.
- Acceso a sitios visitados y comentarios de contactos.
- Visualización de información adicional de los contactos del usuario
- Chat para comunicarse con los contactos.

Conectividad Externa
- Interfaz para comercios para la gestión de promociones u ofertas.
- Reportes, estadísticas y gráficos para los comercios.
- Posibilidad de solicitar taxis desde su ubicación a través de la aplicación.
- Interfaz con las compañías de ómnibus.
- Integraciones con comercios para gestión de reservas, compras, etc.
- Visualización de información demográfica.
- Obtención del pronóstico del tiempo.
- Conexión con otras redes sociales.
- Login con cuenta de Facebook.

Otros

- Guardar contraseña encriptada en base de datos del sistema o utilizar autentificación ldap.
- Implementar notificaciones push que permitan al usuario recibir información sin necesidad de presionar los botones de actualizar.

Uno de los requerimientos que no se pudo desarrollar fue una interfaz grafica web para la administración de sitios debido a limitaciones en el cronograma y redefinición de prioridades a partir de las funcionalidades y requerimientos. Posterior a una intensa investigación de cómo y dónde obtener los puntos de interés, se resolvió manejarlos con una base de datos y un plugin para datos espaciales lo cual no estaba planificado desde el inicio del proyecto. Dicha decisión demostró ser acertada al comprobar cómo aplicaciones externas que ofrecían este servicio de forma gratuita comenzaron a cobrar sin previo aviso al usuario, como el caso de SimpleGeo, uno de los servicios analizados.
Queda también como trabajo futuro cerrar los 5 tickets de prioridad baja y media que no se pudieron resolver por falta de tiempo, de todas formas estos bugs no constituyen un punto crítico en la ejecución de la aplicación.

Capítulo VI. Resultados y Conclusiones

1 Resultados

1.1 Introducción

Se expone un resumen de los principales resultados obtenidos a lo largo del proyecto.

1.2 Testing

Como se indica en el plan de pruebas, se generó un set de casos de pruebas para ejecutar al final de cada iteración. A continuación se muestra un ejemplo de la ejecución de un caso de prueba:

ID CU	Nombre	ID CP	Requisitos	Descripción del caso de prueba	Datos de prueba	Valor esperado	Resultado (ok, error)	Observaciones	Responsable	Validación
CU01	Datos obligatorios	CP01-04		El usuario no ingresa los datos obligatorios.	**e-mail:** (vacio) **contraseña:** (vacio) NOTA: La prueba se deberá ejecutar combinando todas estas opciones.	Mensaje de error indicando que cierto campo no puede ser vacio.	ERROR	El sistema no esta validando esto, solo valida el formato del mail. No se controla si los campos van vacios. Si la contraseña va vacía sale un error no amigable	Martin Loureiro	OK

El total de casos de prueba generados fue de 76, para ver el detalle de los mismos referirse el punto 12 de los anexos.

La ejecución de los casos de prueba generaron un total de 93 incidentes, de los cuales:

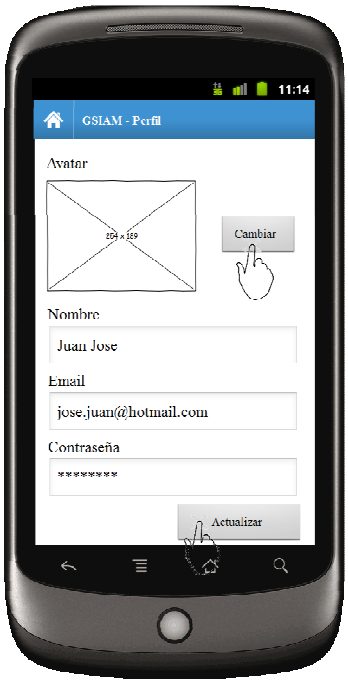
Prioridad	Cantidad
Critical	9
High	9
Medium	63
Low	12

Al final de cada iteración, el equipo realizo pruebas cruzadas de código fuente y documentación. Esto fue importante para encontrar errores de estándares, ortográficos, sintácticos, etc. La rápida detección de dichos errores le permitió al equipo corregirlos mejorando así la calidad del producto final entregado.
A continuación se muestra una revisión técnico formal a modo de ejemplo. Para ver el detalle de todas las revisiones referirse al punto 13 de los anexos.

Fecha:	16/10/2011
Responsable: Tipo Revisión:	Martin Loureiro Documentación
Objeto a Revisar:	Plan de Gestión de Configuración de Software, Estado del Arte
Errores/Observaciones:	
• El punto de Organización del SCM están erróneos los conceptos de “encargado de GCS” y “comité”. • El punto de Seguridad y Respaldo del SCM no se nombra la herramienta de backup. • En el Documento de estado del arte en el punto 3.3 se está usando un tipo de letra Times New Roman en vez de Arial • En el estado del arte la imagen de ciclo de vida de una Activity se ve mal. • En ambos documentos existen varias frases que no respetan el tiempo verbal definido, por ej “Otras de las características que nos inclinaron para elegir a…”. • En ambos documentos existen faltas de ortografía.	
Acciones a Ejecutar:	
• Corregir los conceptos erróneos del SCM, hay que aclarar que no se generara una estructura formal en lo que refiere a la aprobación de los cambios y que no habrá un encargado de SCM. • En el SCM agregar el nombre de la herramienta de backup “Gdocbackup”. • Cambiar tipo de letra a Arial en el punto indicado. • Insertar la imagen indicada de manera correcta. • Corregir el tiempo verbal basándose en el definido para el proyecto en ambos documentos. • Corregir todas las faltas ortográficas.	

1.3 Interfaz Grafica

Durante al análisis y diseño de cada iteración, a demás de generar los diagramas de clase, secuencia y casos de uso, se generaron los prototipos de pantalla correspondientes utilizando Pencil Project [*ref.040*].
A continuación se muestra el prototipo de interfaz grafica del editar usuario y la pantalla real final en el sistema. Para ver todos los prototipos de pantallas generados y las pantallas finales de GSIAM, referirse a los puntos 9 y 11 de los anexos.



Para la interfaz grafica se utilizó GreenDroid es un librería para Android, que facilita el desarrollo de la interfaz grafica además de hacerla mucho más amigable para el usuario. Dentro de sus principales características, se destaca, la inclusión de una barra superior con diferentes funcionalidades configurables que le dan al usuario mayor usabilidad. Para utilizarlo en un proyecto Android, solo basta con incluir el proyecto GreenDroid al workspace y luego incluirlo como librería a nuestro proyecto.

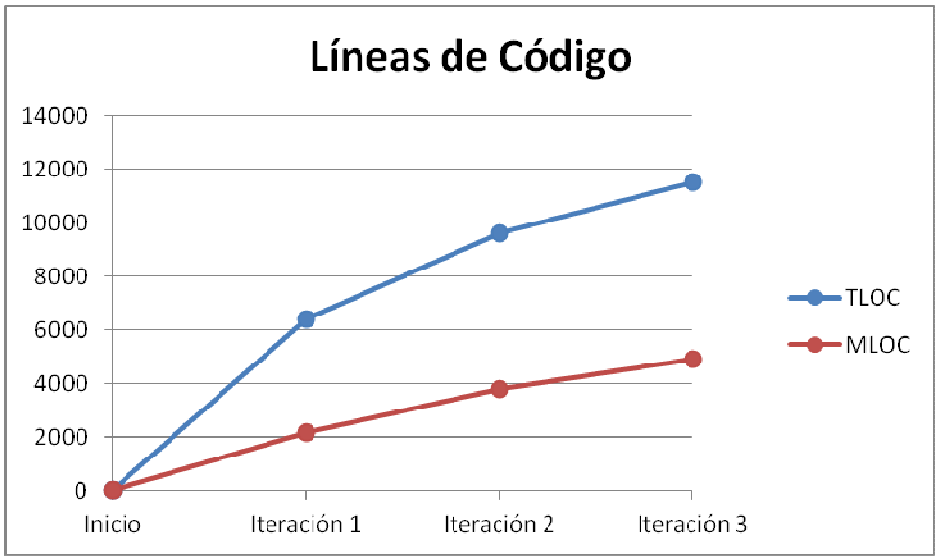
1.4 Metricas

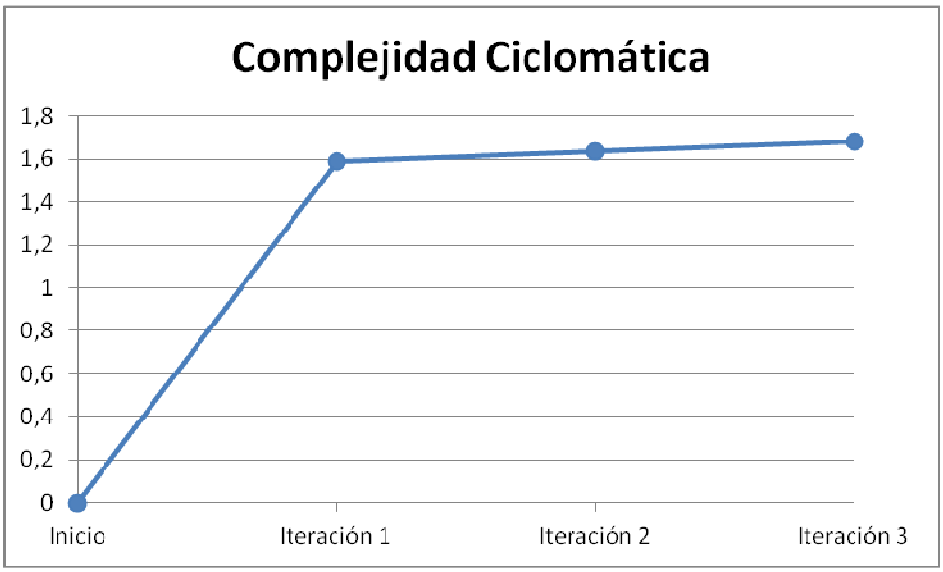
Como se menciono en el Plan de SQA se definieron varias métricas con el objetivo de entender, monitorear y gestionar el proceso de software y con ello adquirir herramientas para mejorar la calidad del mismo.
A continuación se ejemplifican algunas de las métricas, para ver el detalle del análisis de métricas, referirse al punto 14 de los anexos.

1.4.1 Código

La siguiente tabla presenta los valores de las métricas de código correspondientes al final de cada iteración:

Métrica	Iteración 1	Iteración 2	Iteración 3
Líneas de Código (MLOC)	2177	3812	4894
Líneas de Código (TLOC)	6382	9613	11527
Promedio de Métodos por clase	2,92	3,64	4,07
Complejidad Ciclomática McCabe	1,59	1,64	1,68
Profundidad del árbol de herencia	1,71	1,78	1,85





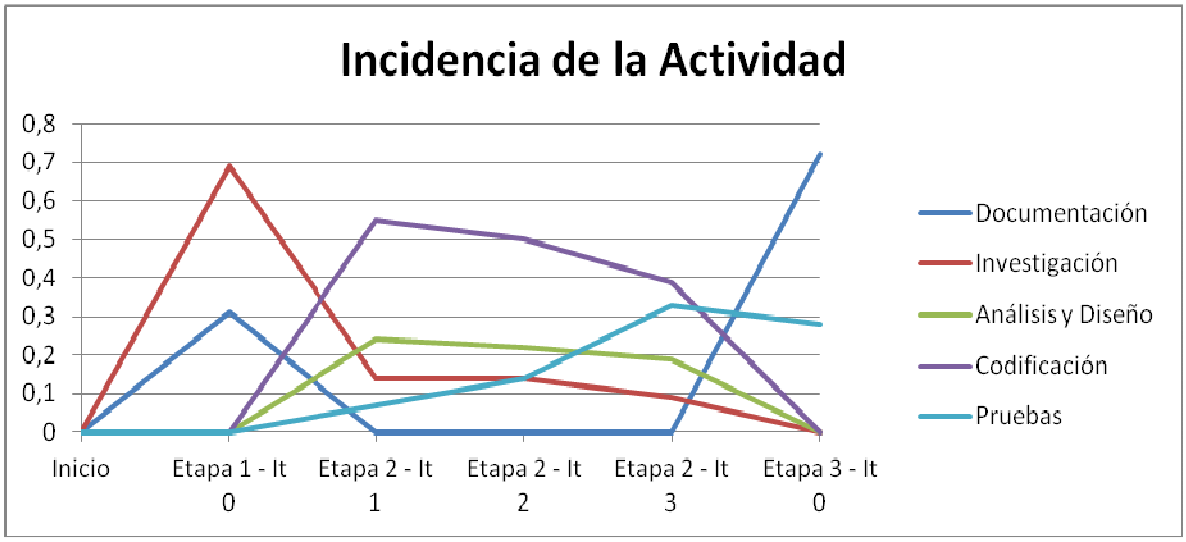
En las métricas por líneas de código y el promedio de métodos por clase se puede observar y comprobar cómo se cumplió lo planificado en el diagrama de gantt en donde las dos primeras iteraciones el mayor esfuerzo se centro en la codificación.

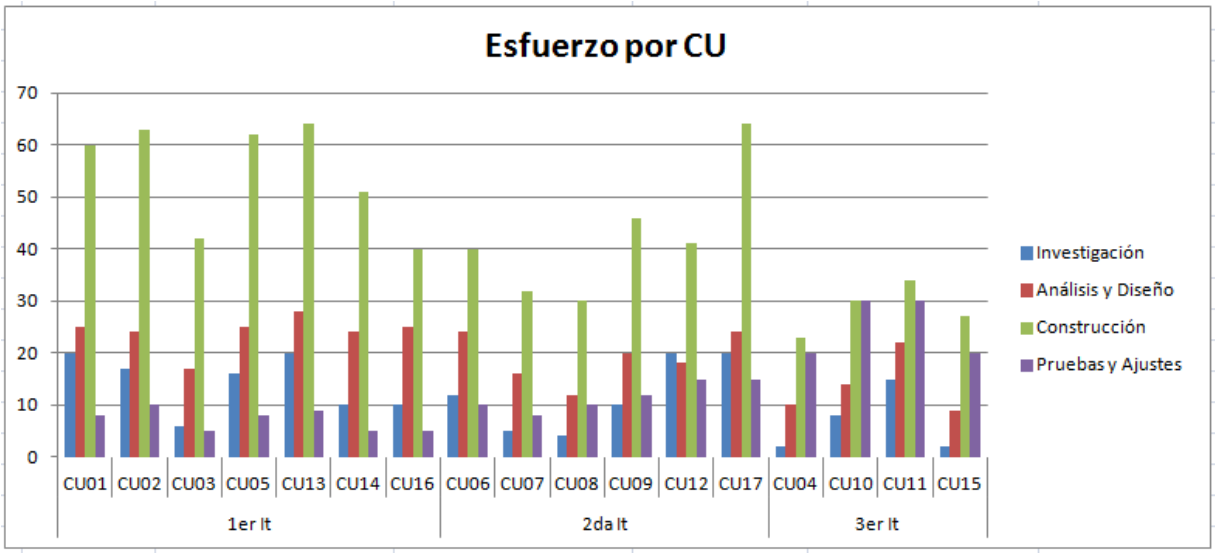
Las métricas referentes a la complejidad del código también arrojaron buenos resultados ya que el mayor crecimiento de estas se da en las iteraciones uno y dos, acompañando a los resultados obtenidos en las métricas por líneas de código.

El sistema se podrá clasificar como un "Programa simple, sin mucho riesgo" debido a que la complejidad ciclomática se encuentra entre 1 y 10.

1.4.2 Esfuerzo

Las siguientes figuras presentan los valores de las métricas de esfuerzo correspondientes al final de cada iteración:





Como se puede ver en las graficas se hizo una separación de esfuerzo por tarea y por casos de uso permitiendo visualizar cuáles fueron los casos de uso más complejos y cuales tareas marcaron la dificultad del proyecto.
El equipo considera que gracias al buen trabajo de investigación realizado se pudieron mantener horas de codificación bajas en relación a las características del proyecto y a la experiencia previa de los integrantes del equipo.

2 Conclusiones

La búsqueda de la propuesta de proyecto de grado fue extensa ya que la intención principal era llevar a cabo un producto de utilidad para la sociedad, con nuevas tecnologías y de código abierto con el fin de incentivar futuros desarrollos.
Cabe destacar que Gsiam posee una arquitectura en base a interfaces que le otorga la flexibilidad necesaria para que pueda ser extendido e incorporarle múltiples funcionalidades en futuras versiones las cuales lo podrían transformarlo en un producto más que atractivo para su futuro desarrollo e incluso podría ser de utilidad para el comienzo de futuros proyectos.
Otra de las ventajas del diseño basado en interfaces y el hecho de que los sitios de interés son administrados por Gsiam mediante servicios, es que se podrían exponer para ser consumidos por terceros sin necesidad de hacer demasiados cambios.
Fueron muchas las dificultades que se presentaron a lo largo del proyecto y que representan el gran desafío que significó su ejecución. Entre ellas, el reto tecnológico a superar debido a que ninguno de los integrantes del equipo tenía experiencia previa en las tecnologías utilizadas. Fue necesario un duro trabajo de investigación y capacitación en el manejo en estas herramientas. Se concluye que los conocimientos

adquiridos en la prueba de concepto fueron de vital importancia para el éxito del proyecto.
Otra de las dificultades a enfrentar durante el proyecto fue la modalidad de trabajo que se implemento dado que uno de los integrantes se encontraba viviendo en el exterior y otro en Montevideo. Esto genero que las distintas herramientas de comunicación actuales hayan cumplido un rol fundamental para que el proyecto haya terminado satisfactoriamente y sin rupturas de lazos entre los integrantes del equipo sino que por el contrario alimentó una mayor unión.
Entre las herramientas que fueron clave para lograr una comunicación fluida cabe mencionar el uso de Google Docs y Skype. Google Docs permitió editar simultáneamente documentos llevando registro de notas y un historial de las mismas para contar una traza de los errores y comentarios. Skype facilito enormemente las reuniones con el tutor, y si bien se tuvieron reuniones presenciales en varias ocasiones con todos los integrantes del proyecto, también se realizaron reuniones virtuales que fueron igual de efectivas.
Fue fundamental el apoyo brindado por el tutor, quien gracias a su amplia experiencia y gran conocimiento técnico respaldo al equipo en todo momento y significó un pilar en el desarrollo del proyecto.
Nos llena de satisfacción poder afirmar que se obtuvieron sólidos conocimientos en las tecnologías, superando así un hito importante en el proyecto de grado.
Queda por ultimo destacar que el producto final obtenido colmó plenamente las expectativas y posee un gran potencial para continuar su desarrollo en futuros proyectos.

Capítulo VII. Apéndice

1 Bibliografía

- Roger Pressman - “Ingeniería del Software, un enfoque práctico”.
- Beginning J2ME - from novice to professional.
- Jim Arlow y Ila Neustadt - UML 2.
- Sergio Gálvez Rojas, Lucas Ortega Díaz - “J2ME Java a Tope” : http://www.lcc.uma.es/~galvez/J2ME.html
- The PostgreSQL Global Development Group - PostgreSQL 9.0.7 Documentation: http://www.postgresql.org/docs/9.0/static/index.html
- Paul Ramsey, Kevin Neufeld, Regina Obe - “PostGis 1.5.3”: http://postgis.refractions.net/download/postgis-1.5.3.pdf
- Paul Ramsey - “Traducción Manual PostGis”: http://postgis.refractions.net/documentation/postgis-spanish.pdf
- Shelly McGowan, Ian Springer: JBoss AS Administration Console User Guide: http://docs.jboss.org/jbossas/6/Admin_Console_Guide/en-US/pdf/Admin_Console_Guide.pdf
- Philippe Kruchten - Architectural Blueprints The “4+1” View Model of Software Architecture:http://www.cs.ubc.ca/~gregor/teaching/papers/4+1view-architecture.pdf

2 Referencias

[*ref.001*] Crecimiento del acceso a internet mediante dispositivos móviles: http://www.prnewswire.com/news-releases/la-gsma-anuncia-que-la-proliferacion-de-equipos-conectados-creara-una-oportunidad-de-us12-billones-para-los-operadores-moviles-en-el-ano-2020-131489458.html(último acceso 21/12/2011)
[*ref.002*] GPS: http://es.wikipedia.org/wiki/Sistema_de_posicionamiento_global (último acceso 21/12/2011)
[ref.003] Issue Tracking: http://code.google.com/p/geolocalizacion-uas/issues/list (último acceso 05/02/2012)
[ref.004] Google Code: http://code.google.com/ (último acceso 05/02/2012)
[*ref.005*] Resumen de la norma IEEE Std-1986: http://standards.ieee.org/findstds/standard/983-1986.html (último acceso 07/08/2011)
[*ref.006*] Estándar de Codificación JAVA: http://java.sun.com/docs/codeconv/ (último acceso 02/12/2011)
[*ref.007*] Plugin de eclipse Metrics: http://metrics.sourceforge.net/ (último acceso 02/12/2011)
[*ref.008*] Wiki GSIAM: http://code.google.com/p/geolocalizacion-uas/wiki/Principal (último acceso 05/02/2012)
[*ref.009*] Open Source: http://es.wikipedia.org/wiki/Código_abierto (último acceso 02/12/2011)
[*ref.010*] Cloud Computing: http://es.wikipedia.org/wiki/Computación_en_la_nube (último acceso 21/12/2011)
[*ref.011*] Subversion: http://es.wikipedia.org/wiki/Subversion_(software) (último acceso 10/12/2011)
[*ref.012*] Gdocbackup: http://code.google.com/p/gdocbackup/ (último acceso 21/12/2011)
[*ref.013*] Android: http://es.wikipedia.org/wiki/Android (último acceso 21/12/2011)
[*ref.014*] Java Mobile: http://www.oracle.com/technetwork/java/javame/javamobile/training/jmesdk/index.html (último acceso 21/12/2011)
[*ref.015*] IOS: http://es.wikipedia.org/wiki/IOS_(sistema_operativo) (último acceso 21/12/2011)
[*ref.016*] IOS: http://developer.apple.com/technologies/ios/ (último acceso 21/12/2011)
[*ref.017*] Windows Phone: http://es.wikipedia.org/wiki/Windows_Phone (último acceso 21/12/2011)
[*ref.018*] IDC - Press Release: http://www.idc.com/getdoc.jsp?containerId=prUS22871611 (último acceso 21/12/2011)
[*ref.019*] Imagen arquitectura Android http://developer.android.com/guide/basics/what-is-android.html (último acceso 02/05/2012)
[*ref.020*] The Developer's Guide: http://developer.android.com/guide/index.html (último acceso 21/12/2011)

[*ref.021*] Ciclo de vida de la activity. Imagen tomada de sitio de Android Developer: http://developer.android.com/guide/topics/fundamentals/activities.html (último acceso 02/05/2012)
[*ref.022*] Porcentaje de mercado: http://www.idc.com/getdoc.jsp?containerId=prUS22871611 (último acceso 21/12/2011)
[*ref.023*] Android Estadísticas: http://www.nielsen.com/us/en/insights/top10s/mobile.html (último acceso 21/12/2011)
[*ref.024*] Simplegeo: https://simplegeo.com/ (último acceso 21/12/2011)
[*ref.025*] Geonames: http://www.geonames.org/ (último acceso 21/12/2011)
[*ref.026*] Datos espaciales: http://es.wikipedia.org/wiki/Base_de_datos_espacial#Puntos (último acceso 21/12/2011)
[*ref.027*] PostgreSQL: http://es.wikipedia.org/wiki/PostgreSQL (último acceso 21/12/2011)
[*ref.028*] PostGis: http://es.wikipedia.org/wiki/PostGIS (último acceso 21/12/2011)
[*ref.029*] Building Web Services with JAX-WS: http://docs.oracle.com/javaee/5/tutorial/doc/bnayl.html (último acceso 21/12/2011)
[ref.030] JAX-WS: http://en.wikipedia.org/wiki/Java_API_for_XML_Web_Services (último acceso 21/12/2011)
[ref.031] JSON: http://es.wikipedia.org/wiki/JSON (último acceso 21/12/2011)
[ref.032] SOAP: http://es.wikipedia.org/wiki/Simple_Object_Access_Protocol (último acceso 21/12/2011)
[ref.033] WSDL: http://www.w3.org/TR/wsdl (último acceso 21/12/2011)
[ref.034] Rest: http://es.wikipedia.org/wiki/Representational_State_Transfer (último acceso 21/12/2011)
[ref.035] Restful: http://www.ibm.com/developerworks/webservices/library/ws-restful/ (último acceso 21/12/2011)
[ref.036] GreenDroid: http://greendroid.cyrilmottier.com/ (último acceso 21/12/2011)
[ref.037] Imagen de greendroid: https://github.com/cyrilmottier/GreenDroid (último acceso 04/04/2011)
[ref.038] Facebook API: http://developers.facebook.com/docs/guides/mobile/ (último acceso 04/04/2011)
[ref.039] Resumen de la norma IEEE Std 830 – 1998: http://www.thinkpdf.com/57/571c146d3cff34ba-download.pdf (último acceso 04/04/2011)
[ref.040] Pencil Project: http://pencil.evolus.vn/en-US/Home.aspx (último acceso 04/04/2011)

3 Glosario de Términos

Android: Sistema operativo basado en Linux para teléfonos móviles.

API (Application Programming Interface): En español, interfaz de programación de aplicaciones, es un conjunto de procedimientos, métodos u otras librerías que ofrecen cierta interfaz para ser utilizadas por otro software.

Arquitectura 4+1: Metodología para la presentación y modelado de la arquitectura de software propuesta Philippe Krutchen, que define 5 vistas concurrentes (Vista Lógica, Vista de Proceso, Vista de Implementación, Vista de Despliegue y la vista de Casos de Uso).

Complejidad Ciclomática de McCabe: Es una métrica de software que nos permite saber que tan complejo es la lógica de un programa.

Could Computing: (computación en la nube) es un paradigma que ofrece servicios de computación a través Internet.

EJB (Enterprise JavaBeans): Es una especificación de la plataforma JEE de Oracle que permite definir una metodología estándar para la implementación de la capa de negocios de una aplicación empresarial.

ERS: Especificación de Requerimientos de Software.

Geolocalización: Término que hace referencia a conocer la ubicación geográfica de un objeto automáticamente.

GIS (Sistema de información geográfica): Es la combinación entre hardware, software y datos geográficos diseñados para manipular la información geográfica con el fin de resolver problemas complejos.

GPS: Sistema de Posicionamiento Global.

IEEE: Instituto de Ingenieros Eléctricos y Electrónicos

J2ME: Java Micro Edition es una especificación de un subconjunto de la plataforma Java orientada a proveer una colección certificada de APIs de desarrollo de software para dispositivos con recursos restringidos. Está orientado a productos de consumo como PDAs, teléfonos móviles o electrodomésticos

JDBC (Java DataBase Connectivity): Es una API que permite la ejecución de operaciones a una base de datos desde el lenguaje de programación Java independientemente del motor de base de datos que se esté utilizando.

JME (Java Micro Edition): Es una especificación de un subconjunto de la plataforma JAVA orientada a dispositivos con capacidad de procesamiento reducido por ejemplo (PDAs, teléfonos móviles o electrodomésticos).

JNDI: Interfaz de Nombrado y Directorio Java (Java Naming and Directory Interface) permite localizar un recurso mediante un nombre.

JPA (Java Persistence API): Es la API de persistencia para la plataforma JEE. Surge para sustituir a los Entity Beans dado que estos eran complejos y difíciles de implementar. Toma muchas ideas del framework de ORM Hibernate.

JSON (JavaScript Object Notation): Es un formato liviano para el intercambio de datos. Su simplicidad está dando lugar a su utilización en remplazo de XML.

Junit: Es un API Java para la implementación de test unitarios para dicha plataforma. Permite automatizar las pruebas del software.

Open Source: (código abierto) es el término con el que se conoce al software distribuido y desarrollado libremente.

PDI: Punto de interés.

REST (Representational Estado Transfer): en español, Transferencia de Estado Representacional, Es un conjunto de principios arquitectónicos por los cuales se diseña servicios web basados en los recursos del sistema. Por su simplicidad está sustituyendo a SOAP y las interfaces basadas en WSDL.

RTF (Revisión Técnica Formal): Es una instancia en la cual se validan todos los entregables según el estándar de calidad definido

SDK: Kit de Desarrollo de Software.

Skype: Aplicación para realizar video llamadas sobre internet.

Smartphone: (Teléfono inteligente), es un término comercial para diferenciar a un teléfono móvil que ofrece más funcionalidades de un teléfono celular común.

SMS (Short Message Service): En español, servicio de mensajes cortos, es un servicio para el envío de mensajes de texto entre teléfonos móviles, teléfonos fijos y otros dispositivos de mano.

SOAP (Simple Object Access Protocol): Es un protocolo estándar para el intercambio de datos entre diferentes procesos mediante XML.

SQL (Structured Query Language): Lenguaje de consulta estructurado, es un lenguaje declarativo para realizar consultas a una base de datos relacional.

SVN (SubVersion): Es un sistema de control de versiones que facilita el desarrollo del software. Es software libre bajo la licencia de APACHE y se lo conoce también como svn por el nombre de la herramienta de línea de comandos.

URL: Uniform Resource Locator en español localizador de recursos uniforme.

WSDL (Web Service Services Language): Es un formato XML que se utiliza para definir servicios web.

XML (Extensible Markup Language): Es un metalenguaje lo que quiere decir que no es un lenguaje sino una forma para definir lenguajes para diferentes necesidades de etiquetas desarrollado por Word Wide Web Consortium (W3C).

ZIP: Formato de almacenamiento sin pérdida utilizado para la compresión de datos como documentos, imágenes o programas.