¨ Sistema de Gestión de Alertas del Buscador Orión para Dispositivos Móviles con Sistema Operativo Android¨

Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas

Autor: Juan Pablo Díaz Ferrer

Tutores:
Ing. Mayra Ortíz Labrada
Ing. Surelys Rodríguez García

“Año 59 de la Revolución”

“La Habana, junio 2017”

Frase


"No se vive celebrando victorias, sino superando derrotas."
"Che Guevara"

Declaración de autoría

Declaro por este medio que yo Juan Pablo Díaz Ferrer, con carné de identidad 92101431904 soy el autor principal del trabajo titulado “Sistema de Alertas del Buscador Orión para Dispositivos Móviles con Sistema Operativo Android” y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____

Firma del autor

Juan Pablo Díaz Ferrer

_____

Firma del tutor

Ing. Mayra Ortíz Labrada

_____

Firma del autor

Ing. Surelys Rodríguez García

Agradecimientos:

Agradezco primero que todo a mi madre por no claudicar a pesar de saber lo terco que soy, por no abandonar en ese año de mi vida en que casi todos dieron la espalda y me miraban con desprecio solo por estar enfermo. Por apoyar cada locura que me pasa por la mente y demostrar que un guerrero no lo hace las derrotas sino la disposición para la lucha.

Agradezco a mi padre por siempre estar presente a pesar de vivir lejos de mí. Por él apoyó, los regaños y por mostrarme en él un paradigma profesional el cual seguir y admirar.

A mi hermano por complacer todos mis gustos y colaborar realizar este y otros sueños realidad.

A mis amigos Addiel, Alejandro , Yandri por siempre estar esperando a que llegue para celebrar. Por entender mi persona y soportar las cosas que le digo cuando me paso de sincero.

A mis tutoras por guiarme y ayudar a materializar este sueño.

A mis compañeros de estudio por la amistad sincera y los buenos momentos que compartimos juntos.

A todos los especialistas del centro CIDI que me ayudaron a pesar de las grandes cargas de trabajo que tenía.

Dedicatoria:

A mi madre y padre que tanto se esforzaron por convertir ese sueño en realidad, por ser siempre mi guía y ejemplo a seguir, por su paciencia y dedicación y por su apoyo incondicional.

A mi hermano que lo adoro, espero siempre ser para ti más que un hermano, un amigo.

Resumen

La Universidad de las Ciencias Informáticas crea el buscador Orión con el objetivo de facilitar la recopilación de información para los usuarios de la web cubana y contribuir de esta manera a alcanzar la soberanía tecnología en el país. Orión posee, como una propiedad de valor, un sistema de alertas mediante el que se informa a los usuarios sobre contenidos indexados de acuerdo a sus intereses, previamente declarados. Dado el gran auge que ha mostrado la telefonía móvil en el mundo moderno, en especial los dispositivos que cuentan con sistemas operativos Android se puede afirmar que la ausencia de un Sistema de gestión de alertas orientado a dispositivos móviles es una pérdida de mercado. Partiendo de esto, en la siguiente investigación se realiza un estudio de las principales características y funcionamiento de sistemas similares existentes con el fin de desarrollar uno propio para Orión. Una vez realizado el estudio de homólogos se obtuvieron características comunes a tener en cuenta en la solución, dígase: contar con una interfaz amigable, organización de las alertas por categorías, creación de *widget* en la pantalla principal, entre otras. La propuesta de solución estuvo guiada por la metodología de desarrollo AUP-UCI, desarrollada bajo tecnología Android con el IDE Android Studio y framework Symfony desarrollado en el IDE Net Beans. Se realizaron diferentes tipos de pruebas al sistema entre las cuales se encuentran pruebas de rendimiento, pruebas funcionales, pruebas de integración y pruebas de compatibilidad. Dichas pruebas fueron realizadas con el objetivo de comprobar su adecuado funcionamiento y cumplimiento de los requisitos planteados concluyendo que el sistema está listo para ser utilizado.

Palabras clave: buscador, motor, alerta, notificación, teléfono, celular, Android.

Índice

Índice de Tablas

Índice de Ilustraciones

Introducción

El avance acelerado de las Tecnologías de la Información y las Comunicaciones (TIC) iniciado en el pasado siglo catalogó al mundo actual y a la humanidad como “Sociedad de la Información y del Conocimiento” o “Era Digital”. La creación de Internet y las facilidades para obtener equipos de cómputo marcó una nueva etapa en la sociedad. Dicho avance proporcionó nuevas vías de comunicación, métodos para la realización de negocios e interacción entre empresas y facilidades a millones de personas para obtener información para diversos fines.

Paralelamente a este avance, se crean un gran cúmulo de aplicaciones web para todo tipo de gestiones *on-line* en diferentes escenarios. Lo anterior trajo consigo un crecimiento de Internet rápido, pero poco estructurado y desorganizado. A raíz de los grandes volúmenes de información publicados en Internet surge la necesidad de realizar búsquedas de información de forma automatizada y así reducir el grado de dificultad que presentan las búsquedas manuales. Los buscadores web o motores de búsqueda fueron una de las respuestas más factibles para esta necesidad.

Se denominan buscadores, motores de navegación o motores de búsqueda, a programas o herramientas interactivas que facilitan la búsqueda y recuperación de información en Internet. Los motores de búsqueda ofrecen formularios para introducir los datos mediante una interfaz de fácil comprensión para el usuario, el mismo puede introducir una palabra clave o frase y recupera una lista de recursos que se corresponde con el criterio indicado. Los motores no pueden cubrir todos los recursos disponibles en Internet, pero muchos contienen referencias a millones de recursos. Los resultados, por tanto, varían de un motor de búsqueda a otro (Rodríguez, 2003).

Actualmente existen numerosos buscadores en Internet. Entre los más usados por los usuarios se encuentran Yahoo!, Google, Bing, el gigante chino Baidu que intenta posicionarse como el buscador más utilizado a nivel mundial y retirar la hegemonía de la que ha gozado Google durante los últimos años. Yahoo! fue altamente valorado por los usuarios durante los años noventa hasta que Google, creado por L. Page y S. Brin, entró en escena en 1997. En realidad, Google no se posicionaría hasta pasados varios años, erigiéndose por la sencillez de su página principal y su innovador algoritmo *PageRank* como el buscador más utilizado por los usuarios. En la actualidad existe una continua lucha de las empresas realizadoras de buscadores por

hacerse con el mercado de los usuarios y colocarse entre los grandes operadores de servicios de búsquedas. Dichas empresas invierten grandes sumas de dinero y apuestan por sofisticados e innovadores servicios extras unidos a la búsqueda. Así Yahoo! alcanzó convertirse en el buscador más utilizado en Estados Unidos durante el mes de agosto de 2013, rompiendo puntualmente la hegemonía ostentada por Google. Otras compañías están apostando muy fuerte en términos económicos y publicitarios para fidelizar nuevos usuarios a utilizar sus buscadores, como el gigante Microsoft con su buscador Bing (Pastor, 2016).

Desde la década de los 80 del pasado siglo, época de la adopción del sistema operativo Microsoft, el uso de la Informática en Cuba estuvo soportado sobre plataformas privativas. A pesar de lo referido, en la actualidad predomina la percepción por parte de las autoridades del país de no apostar por estos sistemas operativos como una solución a largo plazo para el país. De hecho y por muy diversas razones, toda apuesta a futuro que los incluya como principal plataforma operativa para la nación, constituirá un obstáculo para su desarrollo. Uno de los más fuertes motivos que impulsa a la isla para llevar a cabo su proceso migratorio hacia *software* libre lo constituye el tema de la independencia tecnológica. Sobre esta materia se ha pronunciado el Partido Comunista de Cuba (PCC) a través de sus lineamientos 223 y 226, aprobados el 18 de abril del 2011 durante su VI Congreso. Estos lineamientos se pronuncian por “elevar la soberanía tecnológica...” así como “ejecutar inversiones en la industria electrónica y de informática y comunicaciones...” (República de Cuba, 2011).

Cuba se ha planteado la necesidad de migrar de forma gradual y escalonada hacia plataformas operativas y herramientas de trabajo clasificadas dentro de los estándares del *software* libre. Con tal propósito, en abril del año 2004, el Consejo de Ministros acordó la migración al sistema de código abierto, y aunque no quedó establecida una fecha tope, se orientó que el proceso migratorio debía ser continuo y organizado (González, 2012).

Dentro de este proceso de búsqueda de independencia tecnológica se encuentra el sistema operativo Nova, una alternativa de código abierto para solucionar los problemas de dependencia tecnológica hacia los sistemas privativos. La creación de buscadores web como 2x3, Lupa y la plataforma para búsqueda de contenidos unificados llamada C.U.B.A son algunos pasos de avance visibles y funcionales hacia la independencia tecnológica en la isla de Cuba. Además, en la Universidad de las Ciencias Informáticas (UCI) se procede a la implementación y soporte del Buscador Orión, una aplicación web que brinda facilidades de búsquedas para imágenes, páginas web y documentos.

Dada la dinámica de Internet y el gran cúmulo de información, comúnmente los usuarios tienen necesidades de realizar búsquedas sobre un tema específico, intereses personales o categorías a las que se encuentren suscritos. También pueden darse situaciones en las que se desea obtener información que ha sido revelada con cierta frecuencia o nivel de actualización o escenarios en los que se requieren de grandes volúmenes de información y su búsqueda manual resulta muy engorrosa, dado que requieren grandes cantidades de recursos en cuanto a tiempo y personal.

Buscadores web antes mencionados como Google, Yahoo!, Bing y Baidu implementan un servicio en respuesta a esta necesidad llamado Sistemas de alertas. Entre los más populares se encuentra Google Alert. Un Sistema de alertas permite recibir notificaciones que pueden ser enviadas por correo electrónico o como un hipervínculo a usuarios que no necesariamente tienen una cuenta en Gmail (Mena, 2016). Se debe destacar que Google Alert solo ofrece contenidos del propio buscador Google. Las alertas se envían sólo si el nuevo contenido coincide con los términos de búsqueda seleccionados por el usuario. Las notificaciones que ofrece este motor de búsqueda provienen solo de sitios de noticias, lo que significa que se ignoran páginas personales, bitácoras (blogs), entre otros sitios (Mena, 2016).

El Buscador Orión implementa un módulo similar al antes mencionado, el cual mantiene al usuario actualizado de manera automática de toda la información que es indexada por el motor de búsqueda acerca de su tópico. Envía notificaciones mediante correos electrónicos, facilitándole una pequeña descripción y el enlace al contenido. El servicio de alertas es una característica que contribuye a una mejor experiencia de usuario y aporta valor agregado al motor de búsqueda para la recuperación de la información.

StatCounter es una herramienta de análisis de tráfico web perteneciente a la compañía de igual nombre. Esta herramienta brinda estadísticas para cálculos de uso de servicios web y conectividad a Internet. Además, calcula directamente los hits (visitantes únicos) de millones de sitios, con un promedio de 15 mil millones de visitas analizadas mensualmente. Según datos publicados por StatCounter, por primera vez en la historia hay más usuarios accediendo a Internet a través de sus dispositivos móviles que desde su computadora de escritorio o portátil. El tráfico mundial combinado de teléfonos inteligentes y *tablets* alcanzó el 51,2% en octubre del 2015. Es el llamado final para las empresas que dudaron sobre adoptar un enfoque móvil al brindar sus servicios y es poco probable que la tendencia se invierta. En mayo del 2015, Google ya había informado que hay más usuarios realizando búsquedas desde sus teléfonos celulares. Según las mismas estadísticas las computadoras de escritorio todavía son mayoría con el 56,28% del tráfico. Los dispositivos móviles tienen el

41,76% y las *tablets* solo el 1.92%. El 0,04% restante es del acceso a Internet usando consolas de videojuegos. Se tiene una adopción móvil por encima del promedio latinoamericano donde el 65% de los usuarios accede con su computadora de escritorio (Espeso,2015).

En cuanto a sistemas operativos según las estadísticas ofrecidas por comScore, compañía de investigaciones de *marketing* en Internet dedicada a realizar estudios y proporciona datos de *marketing* y servicios para muchas empresas, además brinda seguimiento de los datos de Internet con el fin de estudiar los comportamientos en línea descubrió que Android obtuvo el 52.4% de los clientes del mercado. Apple, con iOS, le sigue los talones con 42.6% y en tercer lugar, está Microsoft con *Windows Phone* y su 3.3%, mientras que otras compañías menos populares dividen el 1.7% restante del mercado de la telefonía móvil.

Grandes empresas en el campo de los buscadores, mediante el estudio de estadísticas similares a las antes planteadas han decidido llevar sus sistemas a la telefonía móvil. Buscadores como Google, además de contar con una interfaz adaptable, cuenta con la aplicación antes mencionada *GoogleAlert* para el subsistema de alertas de igual nombre. Esta aplicación en su versión 2.0 fue desarrollada sobre la tecnología Android, gestiona y mantiene todas las alertas de Google. Usando esta aplicación, se hace posible y fácil de crear, administrar y buscar en Google de una manera estructurada, especificada y automática. Muy recomendada en campos empresariales con el fin de monitorizar las menciones de las instituciones, seguir las tendencias del sector y seguir su comercialización. Ofrece funcionalidades muy interesantes como identificación de plagio y facilidades para encontrar fuentes para publicaciones y presentaciones.

Por otra parte, el buscador Orión brinda facilidades para que los usuarios puedan acceder a los contenidos alojados en la web en dependencia de sus necesidades y utilizando diferentes tipos de dispositivos. Existe un grupo de usuarios que mantienen un sostenido interés sobre temas específicos alrededor de los cuales realizan búsquedas sistemáticamente. El motor de búsqueda Orión constantemente está indexando nuevos contenidos. Cuenta con un mecanismo de alertas orientado a la web que informa, bajo demanda, a sus usuarios cuando se han encontrado nuevos elementos que están relacionados con los temas que son de su interés, utilizando un esquema de periodicidad y un umbral de relevancia o interés.

Analizadas las estadísticas antes citadas se infiere que la telefonía móvil ha conquistado mucho terreno tecnológico en la sociedad moderna. Siendo así su desaprovechamiento una pérdida de oportunidades y de usuarios ya que en la actualidad la mayoría de conexiones a Internet se realizan desde teléfonos inteligentes.

La ausencia de un sistema de alertas para dispositivos móviles obligará al usuario a conectarse a la interfaz web del buscador Orión para gestionar sus alertas. Lo cual sería una desventaja en relación a los competidores en esta área y un desaprovechamiento de nuevas tecnologías en cuanto a mercado.

Teniendo en cuenta la situación problemática descrita anteriormente se enuncia el siguiente **problema a resolver**: ¿Cómo contribuir a que los usuarios de dispositivos móviles puedan gestionar sus alertas en el buscador Orión?

El objeto de estudio de la investigación lo constituyen los procesos para generar alertas en los buscadores web delimitado por el **campo de acción**: Los procesos de gestión de alertas en los buscadores web para los dispositivos móviles con sistema operativo Android.

Para dar solución al problema antes descrito se plantea el siguiente **objetivo general**: Desarrollar un sistema de gestión de alertas destinado a los usuarios de dispositivos móviles con sistema operativo Android.

En el contexto de la investigación se definen los siguientes **objetivos específicos**:

1. Construir los referentes teóricos fundamentales que sustentan la investigación relacionados con el desarrollo de sistemas para la gestión de alertas en buscadores web desde dispositivos móviles con sistema operativo Android.
2. Identificar las funcionalidades del sistema de gestión de alertas del buscador Orión para dispositivos móviles con sistema operativo Android.
3. Implementar las funcionalidades del sistema de gestión de alertas del buscador Orión para dispositivos móviles con sistema operativo Android.
4. Validar las funcionalidades del sistema de gestión de alertas del buscador Orión para dispositivos móviles con sistema operativo Android a través de pruebas de software.

A continuación, se definen las siguientes **preguntas científicas**:

¿Cuáles son los fundamentos teóricos del proceso para generar alertas de los buscadores web para los dispositivos móviles con sistema operativo Android?

¿Cuáles son las características que debe cumplir el sistema de gestión de alertas del buscador Orión para dispositivos móviles con sistema operativo Android?

¿Qué elementos tener en cuenta durante la etapa de implementación del sistema de gestión de alertas del buscador Orión para dispositivos móviles con sistema operativo Android?

¿Cómo comprobar que el sistema desarrollado permite a los usuarios de dispositivos móviles con sistema operativo Android la gestión de sus alertas en el buscador Orión?

Para dar cumplimento a los objetivos específicos y preguntas científicas planteadas con anterioridad, se definen las siguientes **tareas de investigación**:

1. Estudio de la bibliografía existente sobre los sistemas de gestión de alertas en buscadores web desde dispositivos móviles con sistema operativo Android.
2. Estudio del sistema de alertas del buscador Orión.
3. Identificación de las funcionalidades del sistema de gestión de alertas del buscador Orión para dispositivos móviles con sistema operativo Android.
4. Implementación de las funcionalidades del sistema de gestión de alertas del buscador Orión para dispositivos móviles con sistema operativo Android.
5. Validación de las funcionalidades del sistema de gestión de alertas del buscador Orión para dispositivos móviles con sistema operativo Android a través de pruebas de software.

Para dar solución al problema antes planteado es necesario el uso de **métodos de investigación:**

Métodos teóricos:

Analítico-Sintético: utilizado para realizar el análisis de las herramientas, tecnologías y metodologías con el objetivo de identificar aquellas que puedan ser utilizadas en el desarrollo de la investigación, así como el análisis de la información.

Histórico-Lógico: para realizar un estudio sobre la evolución y desarrollo de los sistemas de gestión de alertas existentes teniendo como objetivo detectar las tendencias actuales en el desarrollo de estos sistemas.

Inducción-Deducción: para el análisis de las características del comportamiento de los sistemas de gestión de alertas para llegar a razonamientos aplicables al problema a resolver.

Métodos empíricos:

Modelación: para la representación mediante diagramas de las características de los objetos y relación entre componentes.

Entrevista: realizada a profesores, especialistas y desarrolladores con conocimientos acerca de Orión, haciendo énfasis en los que actualmente se encuentran vinculados directamente al buscador, lo cual aporta una serie de requisitos funcionales importantes para el desarrollo de la aplicación.

El documento está estructurado en: introducción, tres capítulos, conclusiones, recomendaciones, bibliografía y anexos:

Capítulo 1: Fundamentación Teórica del Sistema de Gestión de Alertas del Buscador Orión para Dispositivos Móviles con Sistema Operativo Android: se realiza un estudio de homólogos al Sistema de gestión de alertas. Se efectúa un análisis de las metodologías, herramientas, lenguajes tanto de programación como de modelado, seleccionando cuáles serán los usados en el desarrollo de la investigación.

Capítulo 2: Sistema de Gestión de Alertas del Buscador Orión para Dispositivos Móviles con Sistema Operativo Android: se plantea todo lo relacionado con la caracterización, funciones y capacidades del sistema a desarrollar, así como una descripción y representación de los diagramas utilizados en las fases de análisis y diseño.

Capítulo 3: Implementación y pruebas realizadas al Sistema de Gestión de Alertas del Buscador Orión para Dispositivos Móviles con Sistema Operativo Android: se realizan y documentan las pruebas realizadas a la solución propuesta, así como el análisis a los resultados obtenidos para comprobar el correcto funcionamiento de la aplicación.

Resultados Esperados:

Se pretende que al culminar la investigación el buscador Orión cuente con un sistema de gestión de alertas para los dispositivos móviles con sistema operativo Android, que posibilite a los usuarios la administración de sus alertas.

Capítulo 1. Fundamentación Teórica del Sistema de Gestión de Alertas del Buscador Orión para Dispositivos Móviles con Sistema Operativo Android

1.1 Introducción

Durante el desarrollo de este capítulo se realiza una investigación acerca de sistemas de alertas tanto en el ámbito nacional como internacional. Se exponen criterios valorativos sobre los conceptos fundamentales asociados a la temática, con el objetivo de comprender cómo han evolucionado estos temas y conocer cuáles son las últimas tendencias en el campo de la telefonía móvil. Además, se brindan descripciones acerca de las herramientas, tecnologías y metodología utilizadas para llevar a cabo el desarrollo del Sistema de Gestión de Alertas del buscador Orión para dispositivos móviles con Sistema Operativo Android.

1.2 Conceptos asociados al dominio del problema

Motores de búsqueda: conocidos también como buscadores de Internet, son programas que permiten localizar coincidencias entre la información que existe en la red y la que demanda un usuario. Funcionan rastreando la red de forma periódica y como resultado de la búsqueda se obtiene aquellos recursos web que se ajustan a los criterios de búsqueda introducidos por un usuario (Pastor, 2016).

Sistema de alertas: los sistemas de alertas se encargan de notificar al usuario acerca de los resultados obtenidos para los criterios de búsqueda de su interés ya sea mediante correos electrónicos o cualquier otra vía (Mena, 2016).

Teléfono celular: Denominado celular debido a que el sistema de comunicación que está basado en una división geográfica dentro de la cual se encuentran grupos (células), conformadas de varias antenas de telecomunicaciones. Anteriormente estos dispositivos solamente tenían la función de enviar y recibir llamadas telefónicas, mientras que en algunos casos tenían la función de enviar *(“Small Message Syst*em”) SMS o sistema de mensajes cortos. Sin embargo, actualmente integran una gran gama de funciones que los hace verdaderos dispositivos multifunciones y en gran medida también microcomputadoras (Granado, 2010).

Dispositivos móviles: también conocidos como computadora de mano, son de pequeño tamaño, con algunas capacidades de procesamiento, móviles o no, con conexión permanente o intermitente a una red, con memoria limitada, diseñados específicamente para una función, pero que pueden llevar a cabo otras funciones más generales (Granado, 2010).

1.3 Sistemas homólogos

En la actualidad, existen diversas aplicaciones destinadas a generar alertas. Estas aplicaciones comúnmente son complementos de sitios web destinados a noticias, clima o deportes. Son de gran utilidad y demandadas por los usuarios ya que los mantiene actualizados sin necesidad de revisar los sitios periódicamente. A continuación, se describen cuatro de las más populares en la web destinadas a sistemas Android.

Flipboard

Es considerada una de las aplicaciones con mejor diseño y optimización en Android. Hace que la experiencia al momento de consultar cualquier tipo de información sea fluida, rápida y agradable. Algo que gusta de esta aplicación es que permite seleccionar el tipo de contenido que se puede visualizar. Se puede configurar para que muestre información relativa a tecnología, motor, política, etc. Otra de sus ventajas es que permite guardar el contenido que se elija para visualizarlo después. También permite elegir entre contenidos público o privado (Aguilar, 2016).

Feedly

Es un aglutinador de RSS (archivos generados por los sitios web para compartir contenido) se puede configurar a gusto seleccionando la información que aparezca. Puede crear diferentes temáticas para así poder ver de forma más cómoda los contenidos favoritos. También tiene integración con Google Now, un asistente personal inteligente desarrollado por Google que está disponible dentro de las aplicaciones para móviles de Google para los sistemas operativos Android. La gran ventaja de Feedly es que se actualiza automáticamente, con lo cual no será necesario esperar para leer las noticias, estas irán apareciendo en cuanto se publiquen. Otro de sus puntos fuertes es la interfaz, que hará que se pueda consultar todo desde la misma aplicación, dejando a un lado el sitio web del que provenga la noticia (Aguilar, 2016).

News Republic[1]

Elegida por Google Editor's Choice como una de las mejores aplicaciones de noticias y actualidad en Google Play, ganadora del premio a la mejor aplicación de información por el Mobile *World Congress* 2015. Esta aplicación permite añadir el contenido que se desee a la pantalla de inicio. Esto significa, que se puede acceder con un solo click a toda la información sobre el tema de interés, una idea simple, pero eficaz. Además, permite programar notificaciones para que alerte de contenidos de última hora (Aguilar, 2016).

Andro4all

Permite estar al tanto de *estadísticas,* noticias y todo lo relativo al mundo Android. Además de tener un diseño cuidado, puede añadir *widget*s a la pantalla de inicio y poner alertas para artículos de última hora (Aguilar, 2016).

Valoración sobre los sistemas homólogos

Luego del análisis de los sistemas homólogos se concluye que son aplicaciones privativas por lo cual no se puede obtener ni modificar su código fuente para poder integrarlas al buscador Orión. A pesar de esto, se detectó la existencia de características comunes entre estas aplicaciones, las cuales serán de mucha utilidad para la implementación del Sistema de Gestión de Alertas del Buscador Orión para Dispositivos Móviles con Sistema Operativo Android. A continuación, se muestran algunas de estas características:

- Cuentan con una interfaz amigable y fácil para que el usuario interactúe.
- Las alertas se dividen en categorías en dependencia del área de conocimiento o rama social de la que derivan.
- Poseen funcionalidades que permiten programar notificaciones acerca de un tema especificado por el usuario.
- Cuentan con funcionalidades para crear *widget*[2] en la pantalla principal lo cual evita que el usuario tenga que ejecutar la aplicación directamente.

---

1 Editor's Choice: es un apartado del *Web Market Store* de Android, en el que los editores eligen las aplicaciones que creen que son las más interesantes para los usuarios.

2 Widgets: es una pequeña aplicación o programa, usualmente presentado en archivos o ficheros pequeños que son ejecutados en la pantalla principal del dispositivo para fácil acceso a funciones

- Las actualizaciones se realizan automáticamente en el instante que el usuario se conecta a una red con acceso a Internet.

1.4 Tecnologías de desarrollo y herramientas

En el presente epígrafe se caracterizan las tecnologías y herramientas a utilizar en el proceso de desarrollo del Sistema de Gestión de Alertas del Buscador Orión para Dispositivos Móviles con Sistema Operativo Android. Se exponen las principales ventajas y usos que tendrá cada herramienta para el desarrollo de la aplicación.

1.4.1. Tecnologías de desarrollo

Android

Basándose en las metas que Cuba ha planteado en cuanto a soberanía tecnológica, Android es una tecnología que se ajusta perfectamente a las necesidades del país. Es un sistema operativo libre, gratuito y multiplataforma, además, de su gran aceptación en el mercado y el gran cúmulo de usuarios reflejados en las estadísticas que se exponen en la introducción de la investigación.

Android es un sistema operativo inicialmente pensado para teléfonos móviles, basado en Linux. El sistema permite programar aplicaciones en una variación de Java llamada Dalvik. Proporciona todas las interfaces necesarias para desarrollar aplicaciones que accedan a las funciones del teléfono de una forma muy sencilla en un lenguaje de programación muy conocido como es Java. Junto a la existencia de herramientas de programación gratuitas, hacen que una de las características más importantes de este sistema operativo sea la cantidad de aplicaciones disponibles (Gonzales, 2011).

Entre las características notables se encuentran que:

- La plataforma es adaptable a pantallas más grandes, VGA, biblioteca de gráficos 2D y 3D basada en las especificaciones de *OpenGL*.
- Utiliza *SQLite* para el almacenamiento de datos.
- Emplea *SMS* y *MMS* como vías de mensajería.
- Soporta múltiples tecnologías de conectividad (*GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, LTE y WiMAX*).

---

frecuentemente usadas y proveer de información visual.

- Emplea varios formatos multimedia (AMR, MP3, MIDI, WAV, JPEG, PNG, GIF, BMP...).
- Incluye además soporte para *hardware* adicional (cámaras de fotos, de vídeo, pantallas táctiles, *GPS*, acelerómetros, termómetros, sensores de proximidad y de presión) (Larramendi,2012).

En la siguiente imagen se muestran los componentes principales de la arquitectura Android:

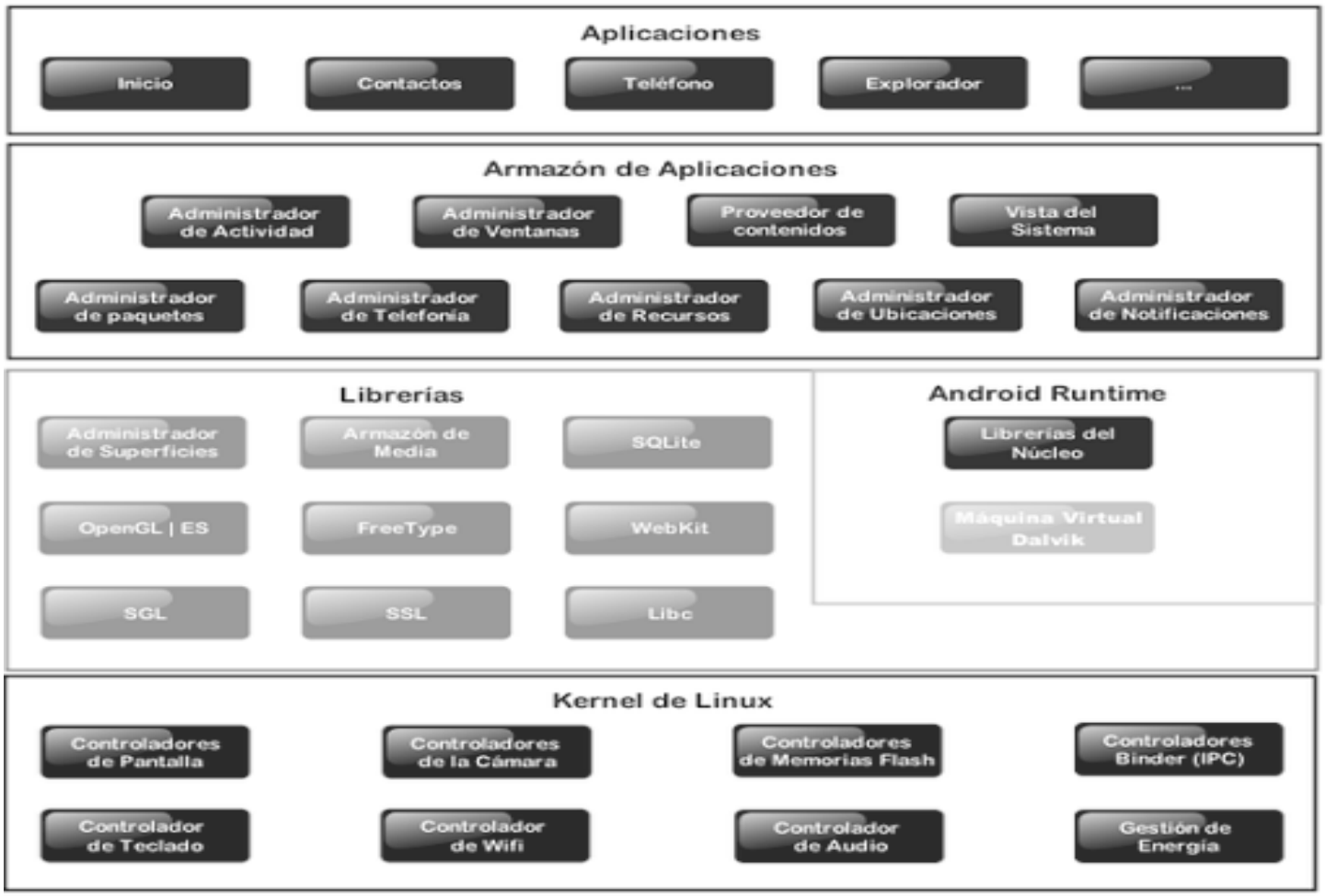


Ilustración 1: Arquitectura Android(Android Developers, 2014).

Android SDK

El SDK del inglés *Software Development Kit* (Paquete de Desarrollo de Software) de Android incluye una variedad de herramientas que ayudan al desarrollo de aplicaciones para la plataforma. Las herramientas son clasificadas en dos grupos: herramientas del SDK y las herramientas de plataformas. Las herramientas del

SDK son independientes de la plataforma y son requeridas sin importar para qué plataforma se está desarrollando la aplicación. Por su parte, las herramientas de plataforma están optimizadas para soportar las características de las plataformas existentes de Android (Android Developers, 2014).

Android SDK incluye:

- Librerías necesarias.
- Entorno de depuración.
- Emulador de dispositivos móviles.
- Documentación relevante para las *APIs* de Android.
- Código fuente de ejemplo.
- Tutoriales para el sistema operativo Android.

Symfony

Es un *framework* diseñado para optimizar el desarrollo de aplicaciones web a través de diversas características claves. Separa las reglas del negocio, la lógica del servidor y las vistas de presentación. Contiene una gran cantidad de clases para conseguir acortar el tiempo de desarrollo en aplicaciones complejas. Adicionalmente, automatiza tareas comunes para que el programador pueda enfocarse por completo en las especificaciones (Rosa, 2013).

Características generales:

- Fácil de instalar y configurar: probado con éxito en plataformas Windows y derivados de Unix.
- Independiente del manejador de Base de datos: utiliza Propel, una capa de abstracción que permite interactuar con varias bases de datos.
- Simple de usar: y al mismo tiempo lo suficientemente flexible para adaptarse a escenarios complejos.
- Cumple con la mayoría de las mejores prácticas de diseño web y patrones de diseño.

- Utilizable en entornos empresariales: puede adaptarse a las políticas y arquitecturas ya existentes en tecnologías de información, y es suficientemente estable para proyectos a largo plazo.

- Fácil de extender: permite integración con otras librerías.

- Manejo de memoria cache: lo cual reduce el ancho de banda y la carga en servidor.

- Mecanismo de autentificación y credenciales: asegura la creación de sesiones restringidas y la gestión de seguridad de usuarios.

- URLs inteligentes: que permiten que las direcciones de las páginas web sean parte de la interfaz y resulten amigables.

1.4.2 Lenguajes de programación

Se definen los siguientes lenguajes de programación pertenecientes a las tecnologías Android y al Framework de desarrollo Symfony sobre el que se encuentra desarrollado Orión.

Java

Es un lenguaje de programación de propósito general, concurrente y orientado a objetos que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Su intención es permitir que los desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo (Steven, 2012).

Características Generales:

- Debido a que existen diferentes productos de Java, hay más de un proveedor de servicios.
- Es independiente de la plataforma de desarrollo.
- Existen dentro de su librería clases gráficas como awt y swing, las cuales permiten crear objetos gráficos comunes altamente configurables y con una arquitectura independiente de la plataforma.

- Java permite a los desarrolladores aprovechar la flexibilidad de la Programación Orientada a Objetos en el diseño de sus aplicaciones.

- El conocimiento sobre tecnología Java está en alto crecimiento en el mercado.
- El sistema de Java maneja la memoria de la computadora. No existe la necesidad de preocupar por apuntadores o memoria que no se esté utilizando.
- El código compilado de Java (conocido como byte code) es interpretado, un programa compilado de Java puede ser utilizado por cualquier computadora que tenga implementado el intérprete de Java.
- Soporta múltiples hilos, puede ejecutar diferentes líneas de código al mismo tiempo.
- No requiere que compiles todas las clases de un programa para que este funcione. Si realizas una modificación a una clase Java se encarga de realizar un Dynamic Bynding o un Dynamic Loading para encontrar las clases (Munguía, 2015).

PHP

(Hypertext Preprocessor) PHP: es un lenguaje de código abierto muy popular especialmente adecuado para desarrollo web y que puede ser incrustado en HTML. En lugar de usar muchos comandos para mostrar HTML (como en C o Perl), las páginas PHP contienen HTML con código incluido en ellas. El código PHP está entre etiquetas de comienzo y final especiales “<? php” y “?>” que permitirán entrar y salir del "modo PHP". Lo que distingue a PHP de lenguajes de ejecución del lado del cliente como Javascript, es que el código es ejecutado en el servidor, generando la fuente HTML y enviándolo al cliente. El cliente recibirá los resultados de ejecutar el script, sin ninguna posibilidad de determinar qué código ha producido el resultado recibido. El servidor web puede ser incluso configurado para que procese todos los archivos HTML con PHP (The PHP Group, 2011).

Entre sus principales ventajas se encuentran:

- Velocidad: está escrito en C, por lo que se ejecuta rápidamente utilizando poca memoria.
- Estabilidad: utiliza su propio sistema de administración de recursos y dispone de un sofisticado método de manejo de variables, conformando un sistema robusto y estable.
- Puede interactuar con muchos motores de bases de datos relacionales tales como MySQL, MSSQL, Oracle, Informix, PostgreSQL y además bases de datos no relacionales como MongoDB.

- Lenguaje de código abierto, lo que posibilita la actualización y corrección de problemas por la amplia comunidad con que cuenta.

- Multiplataforma: permite ser ejecutado bajo varias versiones de Unix, Windows y Mac.

XML

XML, es el estándar de *Extensible Markup Language*. XML, un conjunto de reglas para definir etiquetas semánticas que organizan un documento en diferentes partes. XML es un metalenguaje que define la sintaxis utilizada para definir otros lenguajes de etiquetas estructurados. Dentro de sus características se encuentran (González, 2004):

- Es una arquitectura más abierta y extensible. No necesita versiones para que pueda funcionar en futuros navegadores. Los identificadores pueden crearse de manera simple y ser adaptados en el acto en Internet/intranet por medio de un validador de documentos (parser).
- Mayor consistencia, homogeneidad y amplitud de los identificadores descriptivos del documento con XML (los RDF *Resource Description FrameWork*), en comparación a los atributos de la etiqueta del HTML.
- Integración de los datos de las fuentes más dispares. Se podrá hacer el intercambio de documentos entre las aplicaciones tanto en el propio PC como en una red local o extensa.
- Datos compuestos de múltiples aplicaciones. La extensibilidad y flexibilidad de este lenguaje permitirá agrupar una variedad amplia de aplicaciones, desde páginas web hasta bases de datos.
- Gestión y manipulación de los datos desde el propio cliente web.
- Los motores de búsqueda devolverán respuestas más adecuadas y precisas, ya que la codificación del contenido web en XML consigue que la estructura de la información resulte más accesible.
- Se desarrollarán de manera extensible las búsquedas personalizables y subjetivas para robots y agentes inteligentes. También conllevará que los clientes web puedan ser más autónomos para desarrollar tareas que actualmente se ejecutan en el servidor.
- Se permitirá un comportamiento más estable y actualizable de las aplicaciones web, incluyendo enlaces bidireccionales y almacenados de forma externa (El famoso epígrafe "404 *file not found*" desaparecerá).
- El concepto de "hipertexto" se desarrollará ampliamente (permitirá denominación independiente de la ubicación, enlaces bidireccionales, enlaces que pueden especificarse y gestionarse desde fuera del

documento, hiperenlaces múltiples, enlaces agrupados, atributos para los enlaces, etc. Creado a través del Lenguaje de enlaces extensible (XLL).

- Exportabilidad a otros formatos de publicación (papel, web, cd-rom, etc.). El documento maestro de la edición electrónica podría ser un documento XML que se integraría en el formato deseado de manera directa.

1.4.3. Entorno de desarrollo integrado (IDE)

Eclipse

Fue desarrollado por OTI por sus siglas del inglés *Object Technology International* como un proyecto de IBM17 Canadá. Eclipse es un entorno de desarrollo integrado, de código abierto y multiplataforma para desarrollar proyectos.

Eclipse es una plataforma de desarrollo, diseñada para ser extendida de forma indefinida a través de *plug-ins*. Fue concebida desde sus orígenes para convertirse en una plataforma de integración de herramientas de desarrollo. No está diseñada para un lenguaje específico, sino que es un IDE genérico, aunque es muy popular entre la comunidad de desarrolladores del lenguaje Java usando el *plug-in* JDT que está incluido en la distribución estándar del IDE. Proporciona herramientas para la gestión de espacios de trabajo, escribir, desplegar, ejecutar y depurar aplicaciones.

Principales características

Perspectivas, editores y vistas: en Eclipse el concepto de trabajo está basado en las perspectivas, que no es otra cosa que una pre configuración de ventanas y editores relacionadas entre sí, que permiten trabajar en un determinado entorno de trabajo (Quintero, 2012).

Gestión de proyectos: el desarrollo sobre Eclipse se basa en los proyectos, que son el conjunto de recursos relacionados entre sí, como puede ser el código fuente, documentación, ficheros configuración y árbol de directorios. El IDE proporcionará asistentes y ayudas para la creación de proyectos. Por ejemplo, cuando se crea uno, se abre la perspectiva adecuada al tipo de proyecto que se esté creando, con la colección de vistas, editores y ventanas pre configuradas por defecto.

Depurador de código: se incluye un potente depurador, de uso fácil e intuitivo y que visualmente ayuda a mejorar el código.

Extensa colección de *plug-ins*: están disponibles en una gran cantidad, unos publicados por Eclipse, otros por terceros. Los hay gratuitos, de pago o bajo distintas licencias (Damaian, 2014).

Android Studio

Android Studio es un IDE oficial para el desarrollo de aplicaciones para Android y se basa en IntelliJ IDEA. Además del potente editor de códigos y las herramientas para desarrolladores de IntelliJ, Android Studio ofrece aún más funciones que aumentan la productividad durante la compilación de aplicaciones para Android, como las siguientes:

- Sistema de compilación flexible basado en Gradle.
- Un emulador rápido con varias funciones.
- Un entorno unificado en el que se pueden realizar desarrollos para una gran variedad de dispositivos Android.
- *Instant Run*, para aplicar cambios mientras la app se ejecuta sin la necesidad de compilar un nuevo APK.
- Integración de plantillas de código y GitHub, para ayudar a compilar funciones comunes de las apps e importar ejemplos de código.
- Gran cantidad de herramientas y frameworks de prueba.
- Herramientas Lint para detectar problemas de rendimiento, uso, compatibilidad de versión, etc.
- Compatibilidad con C++ y NDK.
- Soporte integrado para Google Cloud Platform, que facilita la integración *de Google Cloud Messaging* y *App Engine* (AndroidDevelopers, 2014).

Fundamentación del IDE a utilizar:

Después de realizado un análisis acerca de los IDEs antes mostrados (Eclipse y Android Studio) se puede concluir que la alternativa más factible para el desarrollo del Sistema de Gestión de Alertas del Buscador Orión para dispositivos móviles con Sistema Operativo Android es Android Studio. Eclipse consume muchos recursos

en el ordenador, agregando a esto la necesidad de instalar un *plugin* para poder cargar los componentes gráficos de las aplicaciones Android e instalar un emulador externo para poder ejecutar las aplicaciones debido a que no se encuentra integrado, ya que no está inicialmente dirigido al desarrollo de aplicaciones Android. Se necesitará un ordenador con muchas prestaciones en cuanto a *hardware* y de muchas herramientas externas para el desarrollo en el IDE Eclipse. Además, Android Studio está orientado solo al desarrollo de aplicaciones Android. Este sistema consta en su integración con emuladores capaces de imitar a una amplia gama de teléfonos, *tables*, y relojes inteligentes. El formato del código es mostrado de una forma más organizada. No presenta problemas de dependencias ya que tiene un sistema que automáticamente descarga desde los repositorios de Google las librerías necesarias. Cuenta con un grupo de plantillas prediseñadas y una gran facilidad para el diseño de interfaces.

1.5 Metodologías de desarrollo de software

Existe gran cúmulo de información acerca de las metodologías de desarrollo de software y de manera general estas pueden dividirse en dos categorías: tradicionales y ágiles.

Teniendo en cuenta que la universidad se encuentra inmersa en un proyecto de mejora continua y que entre las acciones llevadas a cabo en ese marco se estableció el uso de una misma metodología a aplicar para todos los proyectos, se decide hacer uso de ella. A continuación, se detallan sus principales características:

AUP Variación UCI es, como su nombre lo indica, una adaptación del Proceso Unificado Ágil (AUP) a las características de los proyectos desarrollados en la universidad con el objetivo de estandarizar el proceso de desarrollo en la universidad. Tiene entre sus principales objetivos aumentar la calidad del proyecto final apoyándose en el modelo CMMI-DEV v1.3. El cual constituye una guía para aplicar las mejores prácticas en una entidad desarrolladora.

AUP Variación UCI define tres etapas para el desarrollo de software (Inicio, Ejecución y Cierre). Además, se decide para el ciclo de vida de los proyectos de la UCI tener 7 disciplinas, pero a un nivel más atómico que el definido en AUP. Estas disciplinas son:

1. Modelado de negocio
2. Requisitos
3. Análisis y diseño

4. Implementación
5. Pruebas internas
6. Pruebas de liberación
7. Pruebas de Aceptación (UCI, 2015).

A partir de que el Modelado de negocio propone variantes a utilizar en los proyectos y existen tres formas de encapsular los requisitos, surgen así cuatro escenarios para modelar el sistema en los proyectos.

- Escenario No 1: Si se modela el negocio con Casos de Uso del Negocio (CUN) solo se puede modelar el sistema con Casos de Uso del Sistema (CUS). Este escenario aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan que puedan modelar una serie de interacciones entre los trabajadores del negocio/actores del sistema (usuario), similar a una llamada y respuesta respectivamente, donde la atención se centra en cómo el usuario va a utilizar el sistema.

- Escenario No2: Si se modela el negocio con Modelo Conceptual (MC) solo se puede modelar el sistema con Casos de Uso del Sistema (CUS). Este escenario aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan que no es necesario incluir las responsabilidades de las personas que ejecutan las actividades, de esta forma modelarían exclusivamente los conceptos fundamentales del negocio.

- Escenario No 3: Si se modela el negocio con Descripción de Procesos de Negocio (DPN) solo se puede modelar el sistema con Descripción de Requisitos por Procesos (DRP). Este escenario aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan un negocio con procesos muy complejos, independientes de las personas que los manejan y ejecutan, proporcionando objetividad, solidez, y su continuidad.

- Escenario No 4: Si no se modela el negocio solo se puede modelar el sistema con Historias de Usuarios (HU). Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan un negocio muy bien definido. El cliente estará siempre acompañando al equipo de desarrollo para convenir los detalles de los requisitos y así poder implementarlos, probarlos y validarlos.

Se determina utilizar la metodología AUP Variación UCI en su escenario número cuatro ya que describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software de negocio usando técnicas ágiles. Se ajusta al proyecto ya que está enfocada a las características de los procesos de desarrollo en la

Universidad de la Ciencias Informática al cual pertenece el Sistema de Gestión de Alertas del Buscador Orión para dispositivos Móviles con Sistema Operativo Android. Además, se tiene un dominio completo y específico del proceso de negocio y un contacto directo y frecuente con el cliente.

1.6 Lenguaje de modelado

Lenguaje de Modelado Unificado (UML)

Es la sucesión de una serie de métodos de análisis y diseño orientadas a objetos que aparecen a fines de los 80's y principios de los 90's. UML es llamado un lenguaje de modelado, no un método. Los métodos consisten de ambos, de un lenguaje de modelado y de un proceso. UML incrementa la capacidad de lo que se puede hacer con otros métodos de análisis y diseño orientados a objetos. Los autores de UML apuntaron también al modelado de sistemas distribuidos y concurrentes para asegurar que el lenguaje maneje adecuadamente estos dominios. El lenguaje de modelado es la notación (principalmente gráfica) que usan los métodos para expresar un diseño. El proceso indica los pasos que se deben seguir para llegar a un diseño (González, 2008).

Se usa para diseñar entender, hojear, configurar, mantener y controlar la información sobre los sistemas. Está pensado con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. El lenguaje de modelado pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar. UML incluye conceptos semánticos, notación, y principios generales. Tiene partes estáticas, dinámicas, de entorno y organizativas. Está Pensado para ser utilizado en herramientas interactivas de modelado visual que tengan generadores de código así como generadores de informes (Rumbaugh, James, Jacobson, Ivar y Booch, Grady,2007).

1.6.1 Herramientas CASE

Las herramientas de Ingeniería de Software Asistida por Computadora (CASE) (*Computer AidedSoftware Engineering*) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de *software* reduciendo el costo de las mismas en términos de tiempo y recursos (Davis,1991).

Visual Paradigm (VP):

Constituye una herramienta robusta. Además de que utiliza UML como lenguaje de modelado es colaborativa, es decir, soporta múltiples usuarios trabajando sobre el mismo proyecto. Permite control de versiones y realizar

ingeniería tanto directa como inversa en diferentes lenguajes. Se pueden representar todos los tipos de diagramas UML para las distintas fases del ciclo de vida de un proyecto como la captura de requisitos, análisis, diseño e implementación. Esta herramienta facilita la comunicación, ya que utiliza un lenguaje estándar común a todo el equipo de desarrollo. Presenta la posibilidad de la interoperabilidad con otras aplicaciones como es el *Rational Rose*. Tiene disponible distintas versiones: *Enterprise, Professional, Standard, Modeler, Personal y Community* (que es gratuita). También se facilitan licencias especiales para fines académicos. Por todo lo anterior se selecciona Visual Paradigm 5.3 para realizar el proceso de modelado (Visual Paradigm, 2010).

1.7 Conclusiones parciales

Durante el desarrollo del capítulo se abordaron los elementos correspondientes a la fundamentación teórica de la investigación sobre el Sistema de Gestión de Alertas del Buscador Orión, arribando a las siguientes conclusiones:

- No existe un sistema de gestión de alertas para la telefonía móvil con sistema operativo Android que mantenga a los usuarios del buscador Orión informados de los nuevos contenidos que son indexados.
- El estudio de sistemas similares permitió revelar características de gran utilidad para la interfaz y funcionalidades de la aplicación a desarrollar.
- El estudio de las herramientas y tecnologías permitió definir Android Estudio como entorno de desarrollo para la implementación de la aplicación.
- La elección de la metodología AUP Variación UCI permite tener una mayor organización del proceso de desarrollo de software.

Capítulo 2. Sistema de Gestión de Alertas del Buscador Orión para Dispositivos Móviles con Sistema Operativo Android

2. Introducción

Durante el desarrollo de este capítulo se expone todo lo referente al modelado de negocio, análisis y diseño e implementación del Sistema de Gestión de Alertas del Buscador Orión para Dispositivos Móviles con Sistema Operativo Android. Se especifican los requisitos de *software* que debe cumplir la solución propuesta. Además, se muestran los patrones de diseño utilizados con el fin de lograr buenas prácticas de programación.

2.1 Modelo conceptual

Un modelo conceptual es una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés. Se representa con un conjunto de diagramas de clases en los que no se define ninguna operación. El modelo de domino muestra objetos del dominio, clases conceptuales, asociaciones entre las clases conceptuales y atributos de las clases conceptuales (Craing,2003).

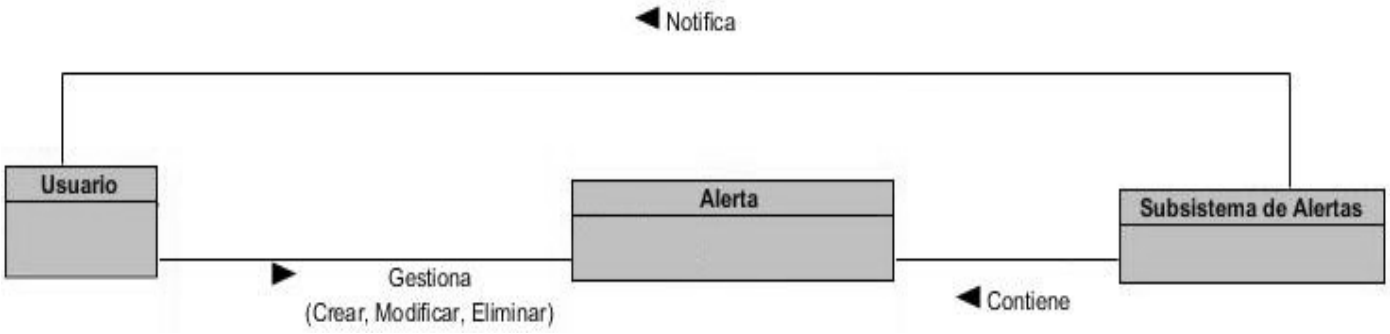


Ilustración 2:Modelo Conceptual

En el modelo conceptual se puede apreciar la existencia de un sistema mediante el cual el usuario puede realizar la gestión de alertas. Después de gestionadas las alertas el Subsistema de Alertas implementado en el buscador Orión es el encargado de realizar el proceso de notificación a los usuarios.

2.2 Requisitos de software

El Institute of Electrical and Electronics Engineers (IEEE)[3] define un requisito como:

1. Una condición o capacidad que un usuario necesita para resolver un problema o lograr un objetivo.
2. Una condición o capacidad que debe tener un sistema o un componente de un sistema para satisfacer un contrato, una norma, una especificación u otro documento formal.
3. Una representación en forma de documento de una condición o capacidad como las expresadas en (1) o en (2).

Se pueden clasificar en funcionales y no funcionales y en ambos casos estos deben ser:

1. Especificados por escrito. Como todo contrato o acuerdo entre dos partes.
2. Posibles de probar o verificar.
3. Descritos como una característica del sistema a entregar.

2.2.1 Requisitos funcionales

Según Ian Sommerville, los requisitos funcionales son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que este debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares. En algunos casos también pueden declarar explícitamente lo que el sistema no debe hacer (Sommerville, 2005).

En la siguiente tabla se muestran los requisitos funcionales de la propuesta de solución:

Tabla 1:Requisitos funcionales

No.	Nombre del requisito	Descripción
RF1	Comprobar estado de red	El sistema debe comprobar si se encuentra conectado a la red para el correcto funcionamiento de la aplicación.
RF2	Comprobar conexión	El sistema debe comprobar si se encuentra conectado con los servicios del *APIRest* implementado en el Buscador Orión.
RF3	Autenticar usuario	EL sistema debe permitir que los usuarios se autentiquen para otorgar los permisos definidos a cada rol.

[3] El glosario estándar IEEE de la terminología de Ingeniería de *Software* identifica los términos actualmente en uso en este campo y la definición estándar de los términos establecidos.

RF4	Mostrar alertas	El sistema debe permitir mostrar todas las alertas creadas por el usuario.
RF5	Insertar alerta	El sistema debe permitir crear nuevas alertas.
RF6	Editar alerta	El sistema debe permitir editar las alertas antes creadas.
RF7	Eliminar alerta	El sistema debe permitir eliminar alertas.
RF8	Buscar alerta	El sistema debe permitir la búsqueda de alertas específicas.

2.2.2 Requisitos no funcionales

Según Ian Sommerville, los requisitos no funcionales son restricciones de los servicios o funciones ofrecidos por el sistema. Incluyen restricciones de tiempo, sobre el proceso de desarrollo y estándares. A menudo se aplican al sistema en su totalidad (Sommerville, 2005).

A continuación, se definen los requisitos no funcionales de la propuesta de solución:

Usabilidad

RNF1. No deben existir enlaces con una profundidad mayor que 3.

RNF2. La interfaz debe ser amigable e intuitiva.

RNF3. El sistema debe mostrar los contenidos de manera organizada por categorías.

Disponibilidad

RNF4. El sistema debe permanecer disponible las 24 horas del día siempre que los servicios del buscador Orión estén disponibles y accesibles.

Seguridad

RNF5. El sistema debe comprobar cada credencial introducida.

RNF6. Las credenciales del usuario no deben enviarse en texto plano.

Software

RNF7. El sistema podrá utilizarse correctamente en dispositivos con sistema operativo: Android 3.0 o superior.

Hardware

RNF8. Para ejecutar el sistema se necesita como mínimo un dispositivo móvil con las siguientes características:

Procesador: 2 núcleos, 1.0 Ghz

RAM: 1GB

Espacio en disco: 20 mb

Conexión de red Wifi o Conexión de datos

2.3. Historias de usuario

Las historias de usuario son el instrumento principal para describir los requerimientos de usuario. Las historias de usuario son descripciones cortas y simples de una funcionalidad, escritas desde la perspectiva de la persona que necesita una nueva capacidad de un sistema, por lo general el usuario, área de negocio o cliente (Rumbaugh y otros,2007). A continuación, se muestran las historias de usuario referentes a los requisitos funcionales Autenticar usuario y Mostrar alertas. Las restantes historias de usuario se pueden visualizar el Anexo I.

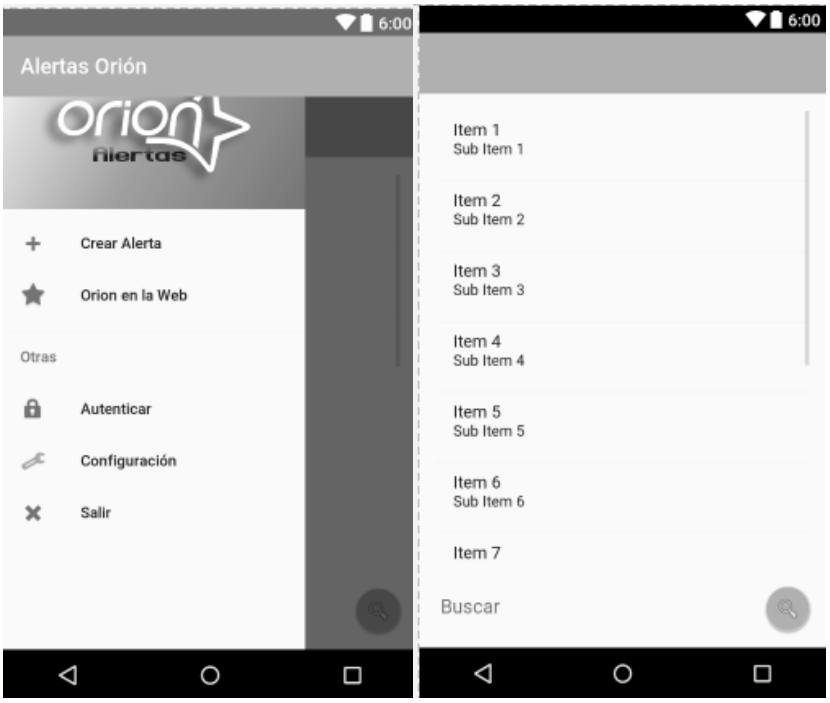
Tabla 2: Historia de usuario Autenticar usuario

Número: 2	Nombre del requisito: Autenticar usuario
Programador: Juan Pablo Díaz Ferrer	Iteración Asignada: 1
Prioridad: Alta	Puntos Estimados: 0.1
Riesgo en Desarrollo: Alto	Puntos Reales: 0.1
Descripción: El sistema mostrará una interfaz con dos campos, uno para introducir la dirección de correo del usuario y otro para su contraseña. Debajo mostrará un botón de aceptar el cual	

después de verificar si son válidos los contenidos introducidos en los campos, si está registrado el usuario y su contraseña es válida, permitirá el acceso a la interfaz principal de la aplicación de lo contrario informará mediante una notificación de fracaso en la autentificación. Observaciones: Al introducir valores no permitidos en el campo de usuario o una contraseña que no cumpla con las políticas de seguridad la aplicación debe mostrar una sugerencia .
Prototipo de interfaz: 

Tabla 3:Historia de usuario Mostrar alertas

Número: 3	Nombre del requisito: Mostrar alertas
Programador: Juan Pablo Díaz Ferrer	Iteración asignada: 1
Prioridad: Alta	Puntos Estimados: 0.1
Riesgo en Desarrollo: Alto	Puntos Reales: 0.1
Descripción: El sistema mostrará una interfaz con todas las alertas que posea el usuario autenticado en su perfil. Al pulsar sobre cualquiera de las alertas abrirá una nueva interfaz con el contenido de la alerta seleccionada. En la parte derecha superior mostrará un botón el cual	

despliega un menú con las funcionalidades para la gestión de alertas y configuración de la aplicación. Observaciones: Al pulsar el botón de la parte superior derecha el listado de las alertas se debe colocar en un segundo plano tomando una opacidad menor a la del menú desplegable.
Prototipo de interfaz: 

2.4. Arquitectura de la propuesta de solución

La arquitectura de *software* se refiere a las estructuras de un sistema, compuestas de elementos con propiedades visibles de forma externa y las relaciones que existen entre ellos (Cervantes, 2015). Para el desarrollo del sistema a implementar se decide utilizar el estilo arquitectónico por capas ya que las aplicaciones Android en su naturaleza utilizan este estilo arquitectónico.

Basado en el sistema propuesto se define la siguiente arquitectura:

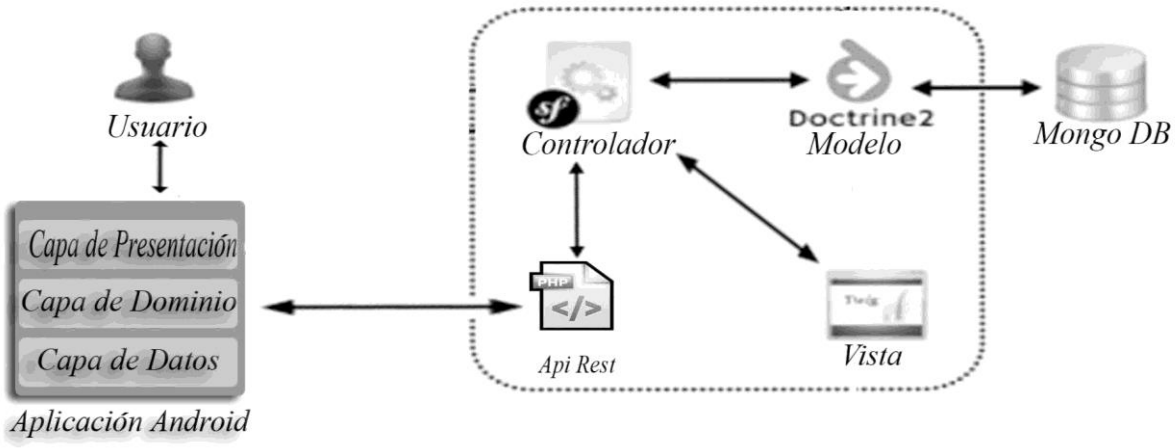


Ilustración 3:Arquitectura del sistema

Como se muestra en la imagen anterior, el sistema que se propone desarrollar está constituido por diferentes componentes, los cuáles serán explicados a continuación:

- Aplicación Android: Interactúa directamente con el usuario.
    - Capa de presentación (Framework Android): en ella se gestiona la lógica relacionada con las actividades, fragmentos y vistas.
    - Capa de dominio: toda la lógica de negocio de la aplicación se gestiona en esta capa, la capa debe ser un módulo de Java puro, sin ninguna dependencia del framenwork de Android. Dentro de ella se implementa la comunicación entre las capas de presentación y datos.
    - Capa de datos: todos los datos que necesita la aplicación proceden de esta capa ya sean servicios web o bases de datos.
- *Api Rest*: Llama las funcionalidades del buscador Orión e interpreta su resultado.
- Controlador: contiene las funcionalidades del buscador Orión:
- Doctrine: gestiona las entidades del buscador Orión.
- Mongo DB: contiene las bases de datos del buscador Orión y ejecuta las consultas.

2.5. Patrones de diseño

Los patrones de diseño son soluciones para problemas típicos y recurrentes que se pueden encontrar al momento de desarrollar una aplicación. Aunque la aplicación sea única, tendrá partes comunes con otras aplicaciones: acceso a datos, creación de objetos, operaciones entre sistemas, etc. En lugar de reinventar los métodos de desarrollo, se pueden solucionar problemas utilizando algún patrón, ya que son soluciones probadas y documentadas por multitud de programadores (Fernández, 2014).

2.5.1 Patrones *GOF*

Los patrones se clasifican según el propósito para el que han sido definidos, a continuación, se seleccionan aquellos patrones *GOF* (Banda de los cuatro) que se adecuan al entorno de desarrollo de aplicaciones compuestas con Android:

- Adaptador (*Adapter*): convierte la interfaz de una clase en otra distinta que es la que esperan los clientes. Permite que cooperen clases que de otra manera no podrían por tener interfaces incompatibles. Dentro de la plataforma móvil, puede emplearse para generar los elementos de componentes visuales como las listas expandibles, que requieren de un adaptador para crear los grupos padres y los hijos que han de ser mostrados al usuario.

  A continuación, se muestra un fragmento de código donde este patrón se evidencia:

```
list = (ListView) findViewById(R.id.alertas);
Alerta x =new Alerta("jupa","si","ddd");
Alerta y =new Alerta("yo","si","ddd");
alertas= new ArrayList<Alerta>();
for(int i=0;i<10;i++){
alertas.add(x);alertas.add(y);}
adapter=new ItemListAdapter(this,alertas);
list.setAdapter(adapter);
adapter.notifyDataSetChanged();
```

Ilustración 4:Patrón Adaptador

- Decorador (*Decorator*): añade dinámicamente nuevas responsabilidades a un objeto, proporcionando una alternativa flexible a la herencia para extender la funcionalidad. Para el entorno de este trabajo, se utiliza de conjunto con el patrón Adaptador, mientras el primero crea los elementos, este le asigna los

atributos correspondientes; por ejemplo, cada vez que una lista expandible se expande o se contrae, el decorador indica qué mostrar en cada índice y subíndice.

A continuación, se muestra un fragmento de código donde este patrón se evidencia:

```
private void inicializar(){
String infService = Context.LAYOUT_INFLATER_SERVICE;
LayoutInflater li = (LayoutInflater)getContext().getSystemService(infService);
li.inflate(R.layout.item_alert_listar, this, true);

lblDescripcion = (TextView) findViewById(R.id.itemlalerta);
 modificar=(ImageButton)findViewById(R.id.itemAlertaModificar);
 eliminar=(ImageButton)findViewById(R.id.itemAlertaEliminar);
//definicion del boton editar
 modificar.setOnClickListener((v) → {
   Intent i= (new Intent(getContext(), ModificarAlertas.class));
   Bundle alertaModificar = new Bundle();
   alertaModificar.putString("nombre",alerta.getNombre());
   i.putExtras(alertaModificar);

   startActivities(getContext(), new Intent[]{i});
 });
```

Ilustración 5:Patrón Decorador

- Comando (*Command*): encapsula una petición en un objeto, permitiendo así parametrizar a los clientes con distintas peticiones, encolar o llevar un registro de las peticiones y poder deshacer las operaciones. En combinación con la tecnología Android, este patrón podrá ser utilizado para encapsular las peticiones a las fachadas de los servicios que consumirán las aplicaciones compuestas.

  A continuación, se muestra un fragmento de código donde este patrón se evidencia:

```
public Conexion(String dir_web, final String Usuario, String Pass) throws MalformedURLException,IOException {
        this.Usuario = Usuario;
        this.Pass = Pass;
        this.dir_web = dir_web;
        this.url = new URL(dir_web);
        this.con = url.openConnection();
        this.aux = con.getInputStream();
    }
```

Ilustración 6:Patrón Comando

- Memento: representa y externaliza el estado interno de un objeto sin violar la encapsulación, de forma que este puede volver a dicho estado en otro momento. En los sistemas móviles se emplea para almacenar el contexto de las vistas junto al estado de las variables en determinado momento, y que el usuario pueda volver a ese estado exacto cuando lo desee.

  A continuación, se muestra un fragmento de código donde este patrón se evidencia:

```
@SuppressWarnings("StatementWithEmptyBody")
@Override
public boolean onNavigationItemSelected(MenuItem item) {
    // Handle navigation view item clicks here.
    int id = item.getItemId();

    if (id == R.id.crear) {
        Intent i =new Intent(this,CrearALerta.class);
        startActivity(i);
        // Handle the camera action
    }
```

Ilustración 7:Patrón Memento

2.5.2 Patrones GRASP

Los patrones *GRASP* constituyen un apoyo para la enseñanza que ayuda a entender el diseño de objetos esencial, y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable. Ayudan a comprender el enfoque para la comprensión y utilización de los principios de diseño basados en los patrones de asignación de responsabilidades (Larman, 2004).

Controlador

Este patrón tiene como objetivo asignar la responsabilidad a una clase de recibir o manejar un mensaje de evento del sistema generado por un actor externo, por lo general a través de una interfaz gráfica de usuario a la que accede un usuario para realizar ciertas operaciones en el sistema (Larman, 2004).

A continuación, se muestra un fragmento de código donde este patrón se evidencia:

```
if(conexion.getRespuesta() != null ) {
    JSONObject aux = new JSONObject(conexion.getRespuesta());
    if (aux.getString("mensaje").equals("Credenciales invalidas")) {
        Snackbar.make(view, aux.getString("mensaje"), Snackbar.LENGTH_LONG)
                .setAction("Action", null).show();
    } else {
        Snackbar.make(view, aux.getString("mensaje"), Snackbar.LENGTH_LONG)
                .setAction("Action", null).show();
```

Ilustración 8: Patrón Controlador

Experto en información

La solución que propone este patrón es la de asignar una responsabilidad a la clase (experto) que cuenta con la información necesaria para cumplirla. Ofrece una analogía con el mundo real ya que da origen al diseño donde el objeto de software realiza las operaciones que normalmente se aplican al elemento real que representa (Larman, 2004).

A continuación, se muestra un fragmento de código donde este patrón se evidencia:

```
public void readstrem(InputStream in){
    this.respuesta="";
    String line ="";
    BufferedReader reader=null;
    reader = new BufferedReader(new InputStreamReader(in));
    try {
        while ((line=reader.readLine())!= null ){
            this.respuesta= this.respuesta+line;
        }

    }
    catch (IOException e){e.printStackTrace();}
}
```

Ilustración 9: Patrón Experto en Información

Creador

Guía la asignación de responsabilidades relacionadas con la creación de objetos. Una de las consecuencias de usar este patrón es la visibilidad entre la clase creada y la clase creador. Una ventaja es el bajo acoplamiento, lo cual supone facilidad de mantenimiento y reutilización La creación de instancias es una de las actividades más comunes en un sistema orientado a objetos. En consecuencia, es útil contar con un principio general para la asignación de las responsabilidades de creación. Si se asignan bien, el diseño puede soportar un bajo acoplamiento, mayor claridad, encapsulación y reutilización (Larman,2004). A continuación, se encuentra un fragmento de código de la clase Autenticar la cual posee los parámetros necesarios para crear un objeto de la clase Conexión.

A continuación, se muestra un fragmento de código donde este patrón se evidencia:

```
EditText usuario =(EditText)findViewById(R.id.usuario);
EditText pass = (EditText)findViewById(R.id.pass);

Conexion conexion = new Conexion("http://"+confiPreferences.getString("url","")+"login/rest?usuario="
        +usuario.getText().toString()+ "&password="+pass.getText().toString());
System.out.println("--------------"+confiPreferences.getString("url",""));
try {
    conexion.conectar("POST");
} catch (IOException e) {
    e.printStackTrace();
```

Ilustración 10: Patrón Creador

Alta Cohesión

Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. Significa que las clases del sistema tienen asignadas solo las responsabilidades que les corresponde y mantienen una estrecha relación con el resto de las clases (Larman,2004).

A continuación, se muestra un fragmento de código donde este patrón se evidencia:

```
}
try {
    JSONArray jsonArray = new JSONArray(conexion.getRespuesta());
    conexion.cerrar();
    alertas= new ArrayList<Alerta>();

    for (int i = 0; i < jsonArray.length(); i++) {
        JSONObject jsonObject = jsonArray.getJSONObject(i);

        String nombre = jsonObject.getString("name");
        String id = jsonObject.getString("id");

        alertas.add(new Alerta(nombre,id));}
```

Ilustración 11: Patrón Alta Cohesión

Bajo Acoplamiento

Determina el nivel de dependencia de una clase con respecto a otras. Una clase con bajo acoplamiento no depende de muchas otras. En los lenguajes orientados a objetos como C++, Java y Smalltalk una de las formas más comunes de acoplamiento de TipoX a TipoY es cuando TipoY es una interfaz y TipoX la implementa (Larman,2004).

A continuación, se muestra un fragmento de código donde este patrón se evidencia:

```
try {
    conexion.conectar("");
    if(conexion.getRespuesta()== null)
    {
        Intent i = (new Intent(MainActivity.this, Autenticar.class));
         finish();
        startActivity(i);
    }
```

Ilustración 12: Patrón Bajo Acoplamiento

2.6. Diagrama de clases

Un diagrama de clases es una representación gráfica que sirve para representar la estructura de un sistema que será implementado utilizando un lenguaje orientado a objetos. Los diagramas de clases se realizan en la fase de diseño del software después de la fase de requisitos. Representa las clases que tendrá el sistema, así como su contenido y sus relaciones con otras clases (Craig, 2003). A continuación, se muestra el diagrama de

clases del Sistema de Gestión de Alertas del Buscador Orión para Dispositivos Móviles con Sistema Operativo Android.

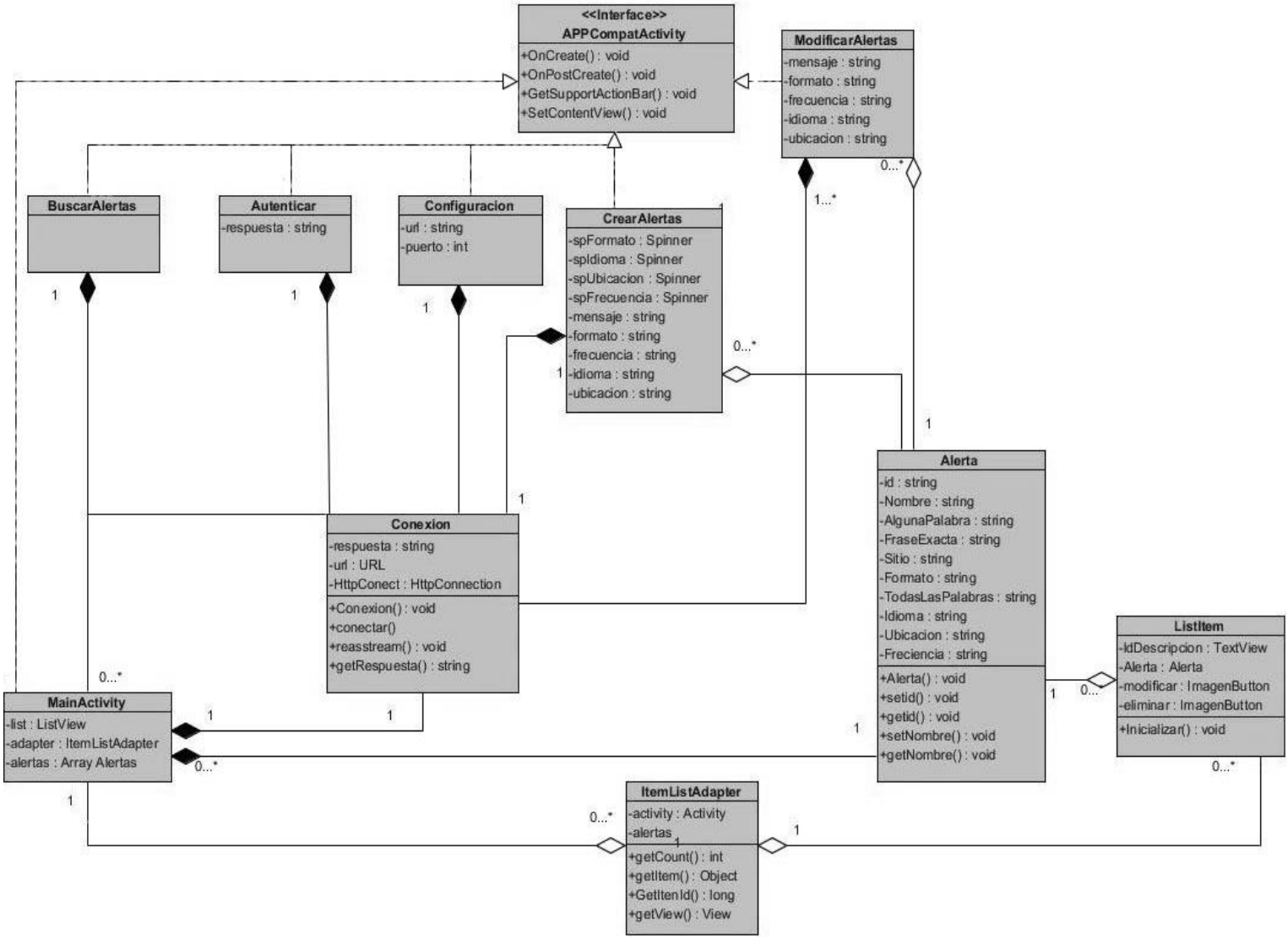


Ilustración 13: Diagrama de Clases

En el diagrama anterior se muestran las clases que interactúan en el Sistema de Gestión de Alertas del Buscador Orión para Dispositivos Móviles con Sistema Operativo Android. Entre estas se encuentran las clases ModificarAlertas, *Main Activity*, *CrearAlerta, BuscarAlertas* y Autenticar las cuales heredan de la clase *APPCompatActivity*. Estas clases contienen los componentes y la programación perteneciente a las vistas de

la aplicación. La clase *Alerta* contiene los atributos y los métodos para realizar operaciones con las alertas que serán gestionadas en la aplicación. La clase conexión contiene todo lo referente al proceso de conexión con el servicio rest implementado en el buscador Orión. Además se evidencias clases auxiliares como *ListItem* y *ItemListAdapter* las cuales son complementos visuales que son utilizados en la clase *MainActivity*.

2.7. Modelo de datos

A continuación, se muestra el modelo de datos físicos de la aplicación, el cual es un esquema de diseño para activos de información que define las estructuras físicas y las relaciones de los datos dentro de un dominio o aplicación específica (Craing, 2003). El modelo de datos físicos es un modelo específico que representa objetos en las bases datos y sus relacionales sin tener en cuenta la estructura funcional de la aplicación.

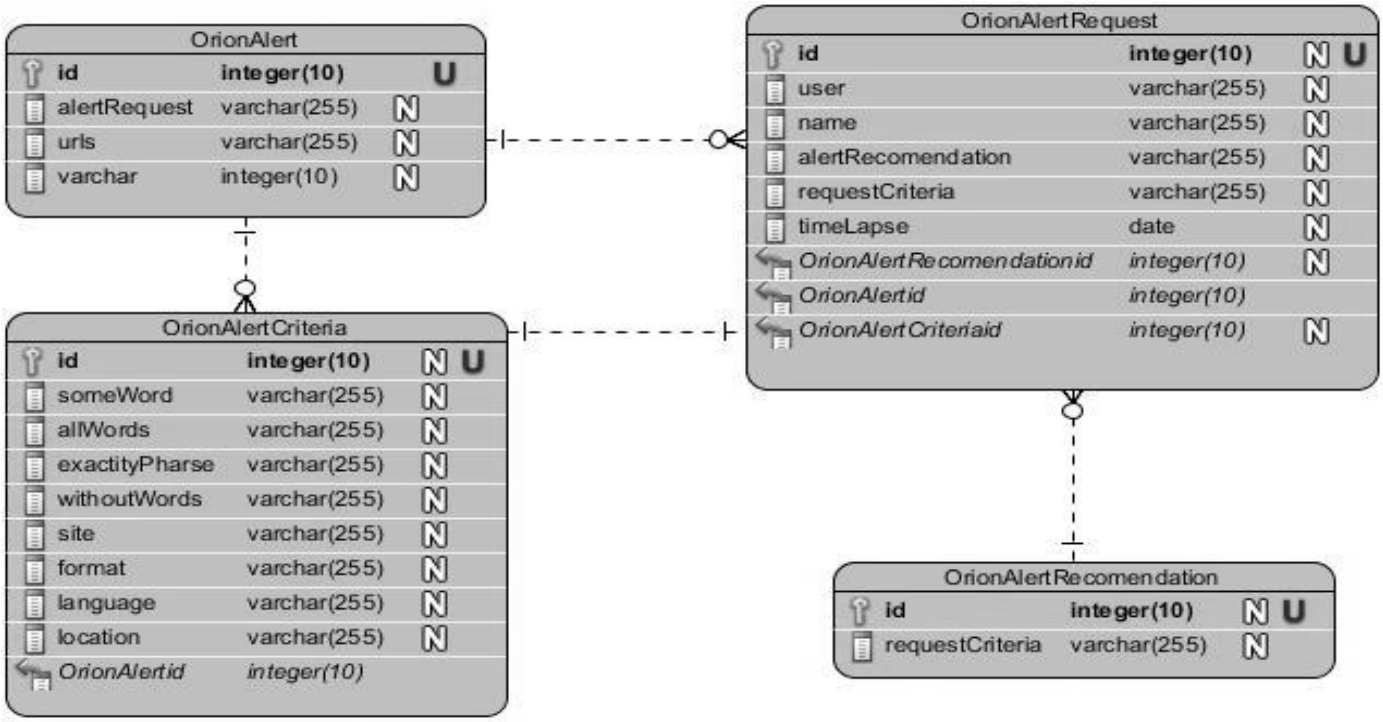


Ilustración 14: Modelo de Datos

2.8. Modelo de despliegue

Los modelos de despliegue son los complementos de los diagramas de componentes que, unidos, proveen la vista de implementación del sistema. Describen la topología del sistema, la estructura de los elementos de

hardware y el *software* que ejecuta cada uno de ellos. Los diagramas de despliegue representan a los nodos y sus relaciones. Los nodos son conectados por asociaciones de comunicación tales como enlaces de red (Craing, 2003).

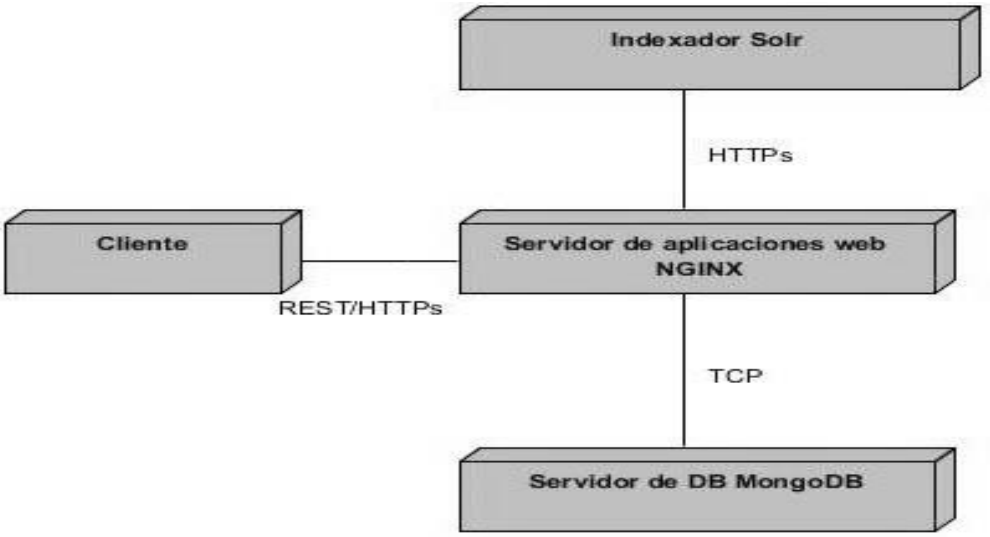


Ilustración 15: Modelo de Despliegue

El diagrama de despliegue propuesto está compuesto por cuatro nodos que se explican a continuación:

- Cliente (dispositivo Android): ejecuta la aplicación Sistema de gestión de alertas que se comunica con el *APRest* implementada en el buscador Orión mediante servicios *REST* y protocolo *HTTPs.*
- Servidor de aplicaciones web: se encuentran las funcionalidades del buscador. El servidor de aplicaciones web es el que permite que el cliente interactúe y tenga acceso al buscador Orión.
- Indexador: Se encarga de realizar la búsqueda de la información por la que el usuario desea ser notificado.
- Servidor de base de datos: guarda los datos correspondientes a las solicitudes realizadas por el usuario.

2.9 Conclusiones parciales

Durante el desarrollo del presente capítulo se alcanzó una mayor comprensión del Sistema de Gestión Alertas del buscador Orión para Dispositivos Móviles con Sistema Operativo Android. Se definieron detalladamente las características con las que debe cumplir y los requisitos a tener en cuenta durante la etapa de implementación. Basándose en estas actividades se arriba a las siguientes conclusiones:

- Los requisitos funcionales y no funcionales brindan una profunda caracterización de la aplicación y las funciones que esta debe realizar.
- El Modelo de datos brinda mayor comprensión de las entidades y sus relaciones en el funcionamiento de la aplicación.
- El uso de patrones *GOF* brinda una mayor compresión de las buenas prácticas de programación y colabora el desarrollo de un código más legible.

Capítulo 3. Implementación y pruebas realizadas al Sistema de Gestión de Alertas del Buscador Orión para Dispositivos Móviles con Sistema Operativo Android

3.1 Introducción

En el presente capítulo se describe la implementación de la propuesta de solución guiada por los requisitos funcionales y no funcionales anteriormente definidos. Se ejemplifican los estándares de codificación, según el lenguaje propuesto. Se plantea la estrategia de validación con las pruebas de software ejecutadas y los resultados obtenidos en cada caso.

3.2. Diagrama de componentes

El diagrama de componentes muestra las interacciones y relaciones de los componentes de un modelo. Entendiéndose como componente una clase de uso específico, que puede ser implementada desde un entorno de desarrollo, ya sea de código binario, fuente o ejecutable; dichos componentes poseen tipo, que indican si pueden ser útiles en tiempo de compilación, enlace y ejecución. Este tipo de diagrama se representa mediante componentes unidos mediante relaciones de dependencia (Rumbaugh, 2007).

En el diagrama siguiente se muestran todos los componentes que interactúan en el Sistema de Gestión de Alertas del Buscador Orión para Dispositivos Móviles con Sistema Operativo Android, entre los cuales se encuentran:

- El paquete Interfaces agrupa los *Recursos Gráficos, Interfaces(Activity)* y librerías usadas para la renderización de los elementos visuales. Este componente almacena todo lo referente a la visualización de la aplicación, donde se encuentran elementos como imágenes, gráficos, menús y ventanas flotantes.

- El paquete Alertas Orión almacena lo referente al funcionamiento de la aplicación, toda la lógica del negocio, dentro se encuentran las entidades que interactúan en la aplicación como la clase Conexión encargada de la realización de la conexión con el *APIRest*. También se encuentran librerías necesarias para el funcionamiento y la entidad Alerta.

- El paquete Orión hace referencia al buscador Orión en si totalidad. Especificando en el diagrama la utilización del *APIRest* y de los controladores correspondientes a esta para las operaciones en el servidor web. Además, se muestran las entidades con las cuales interactúa la aplicación para las gestiones de alertas dentro del dicho buscador.

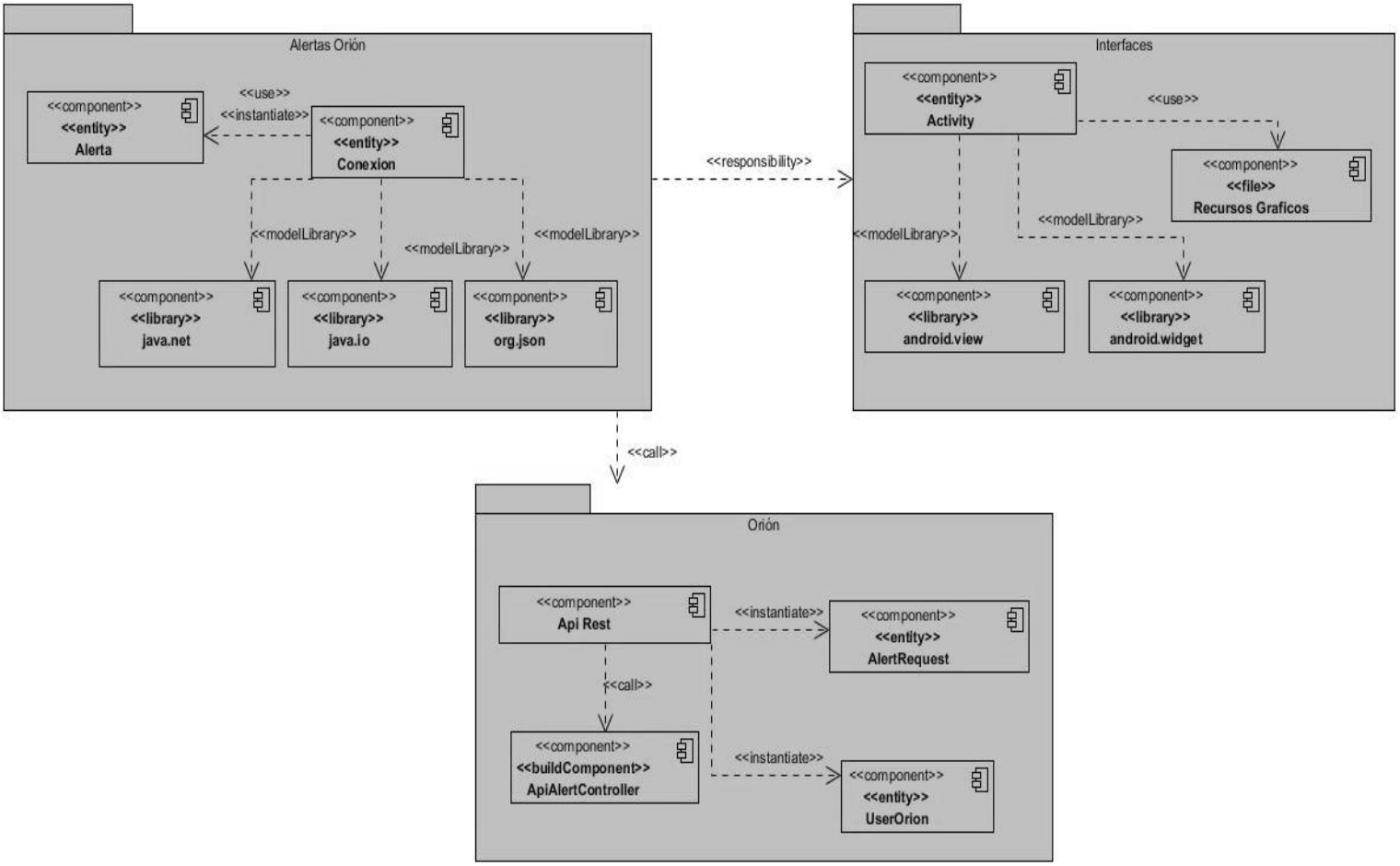


Ilustración 16: Diagrama de Componentes

3.3.1 Estándares de nomenclatura y codificación utilizada para android

Nomenclatura general:

- ✓ El lenguaje por defecto para dar sentido funcional a las clases, variables y constantes será una mezcla entre nomenclatura tradicional en inglés y nomenclatura funcional adoptada.
- ✓ El nombre de todas las variables comenzará con minúsculas, en casos donde el nombre esté compuesto por más de una palabra, todas las palabras que lo componen comenzarán con mayúsculas.

Paquetes:

- ✓ Por defecto todos los nombres de paquetes se escribirán en minúsculas y sin utilizar caracteres especiales.

Identación:

- ✓ En el contenido siempre se adentrará con tabulaciones, nunca utilizando espacios en blanco.

Clases:

- ✓ Los nombres de clases deben ser mezclas de mayúsculas y minúsculas, con la primera letra de cada palabra interna en mayúsculas (CamelCase).
- ✓ Se debe intentar mantener los nombres de clases simples y descriptivos.
- ✓ Se deben usar palabras completas y evitar acrónimos y abreviaturas.

Métodos:

- ✓ Los métodos deberán ser verbos (en infinitivo), en mayúsculas y minúsculas con la primera letra del nombre en mayúscula y con la primera letra de cada palabra interna en mayúsculas.
- ✓ No se permiten caracteres especiales.
- ✓ El nombre ha de ser lo suficientemente descriptivo, no importando a priori la longitud del mismo.

Variables:

- ✓ El nombre de las variables debe comenzar con letra minúscula y de existir un salto de palabra comenzaría con mayúscula, en el caso de los atributos de las clases, estos comenzarán con guión bajo.

Constantes:

- ✓ Los nombres de constantes deben escribirse todo en mayúsculas con las palabras separadas por guión bajo. Todas serán declaradas como *public static* final.

Comentarios:

- ✓ Siempre se escribe en tercera persona.
- ✓ Las descripciones siempre deberán empezar por un verbo.

3.3.2 Estándares de nomenclatura y codificación utilizada para Symfony

Estructura

- ✓ Hacer uso de las etiquetas cortas (<?).
- ✓ Finalizar las clases con la etiqueta usual de cierre (?>).
- ✓ La indentación se debe realizar utilizando cuatro espacios.
- ✓ Agregar un único espacio después de cada delimitador coma.
- ✓ No poner espacios después de la apertura de un paréntesis y antes del cierre del mismo.
- ✓ Agregar un único espacio alrededor de operadores (==, &&, ...).
- ✓ Agregar un único espacio antes de los paréntesis de apertura de una palabra clave de control.
- ✓ Agregar una línea en blanco antes de la sentencia *return*.
- ✓ No agregar espacios al final de las líneas.
- ✓ Utilizar llaves para indicar el cuerpo de las estructuras de control sin importar el número de sentencias que estas contengan.
- ✓ Colocar las llaves en sus propias líneas para clases, métodos y declaración de funciones.
- ✓ Separar las sentencias condicionales y las llaves de apertura con un único espacio sin dejar una línea en blanco.
- ✓ Declarar explícitamente la visibilidad de clases, métodos y propiedades (el uso de var está prohibido).
- ✓ Utilizar constantes de tipo PHP nativas en minúsculas: *false, true* y *null*. Igual aplica para *array()*.

- ✓ Utilizar letras mayúsculas para constantes, con palabras separadas por guiones bajos;
- ✓ Definir una clase por archivo.
- ✓ Declarar las propiedades de las clases antes de los métodos.
- ✓ Declarar los métodos públicos primero, luego los protegidos y finalmente los privados.

Convención de nombres

- ✓ Utilizar *camelCase* y no guiones bajos, para variables, funciones y nombres de métodos.
- ✓ Utilizar guiones bajos para definir opciones, argumentos y nombres de parámetros.
- ✓ Utilizar los *namespace* para todas las clases.
- ✓ Anadir como sufijo *Interface* a las interfaces.
- ✓ Utilizar caracteres alfanuméricos y guiones bajos para nombres de archivos.

Documentación

- ✓ Agregar los bloques *PHPDoc* para todas las clases, métodos y funciones.
- ✓ Las anotaciones *@package* y *@subpackage* no son utilizadas.

3.4 Pruebas del sistema

La realización de pruebas permite comprobar la eficiencia del sistema y verificar el cumplimiento de los objetivos trazados a lo largo del ciclo de vida del proyecto. Mediante la realización de pruebas se somete la aplicación a situaciones o ambientes similares a los que funcionará en un futuro, lo cual ayuda a reducir errores, aumentar la seguridad y conocer los tiempos de respuesta del sistema.

3.5.1 Estrategia de prueba

Según Pressman una estrategia de prueba de software proporciona una guía que describe los pasos que deben realizarse como parte de la prueba y (...) debe ser suficientemente flexible para promover un uso personalizado de la prueba (Pressman,2010).

Para la validación de la propuesta de solución se llevaron a cabo las pruebas que se describirán a continuación.

3.5.1 Pruebas de caja negra

Se centran en los requisitos funcionales, son pruebas realizadas al producto final. Están enfocadas en las entradas, las salidas de la aplicación y la relación entre ellas. Estas pruebas se realizan poniendo en práctica

las funcionalidades y comprobando que los resultados obtenidos sean igual a los resultados esperados (Gonzales,2012).

La variante de caja negra es la utilizada para probar el sistema, dentro de ella existen varias técnicas, una de las más prácticas y eficaces es la conocida como “partición de equivalencia”. Esta divide la entrada en clases de datos según las funcionalidades a las que tributan en el sistema, brindando así la posibilidad de descubrir funcionalidades incorrectas o ausentes, errores en la interfaz, de rendimiento, de inicialización o de terminación.

A continuación, se muestran los casos de prueba para los requisitos Autenticar usuario y Mostrar alertas**:**

Condiciones de Ejecución:

- ✓ Conexión a Internet.
- ✓ Los servicios del buscador Orión deben estar disponibles.

Tabla 4: Caso de prueba autenticar usuario

Caso de prueba autenticar usuario			
Nombre de Sección	**Escenario de Sección**	**Descripción de funcionalidad**	**Flujo central**
NS1:Autenticar usuario	ES1: Autenticación satisfactoria	El sistema muestra una interfaz con las alertas correspondientes al usuario y las principales funcionalidades de la aplicación	- Ejecutar la aplicación en el teléfono - Introducir los datos en el formulario que muestra la aplicación - Seleccionar Aceptar
	ES2: Datos introducidos incorrectos	El sistema muestra un mensaje flotante informando que el usuario o la	- Ejecutar la aplicación en el teléfono - Introducir los datos en el formulario que muestra la aplicación

		contraseña no es correcto.	- Seleccionar Aceptar
	EC3: Ausencia de conexión a Internet	El sistema muestra un mensaje flotante informando que se necesita estar conectado a Internet para realizar la operación.	- Ejecutar la aplicación en el teléfono - Verificar la conexión a Internet - Introducir los datos en el formulario que muestra la aplicación - Seleccionar Aceptar.
	EC4: Seleccionar opción Atrás	El sistema debe finalizar todas las operaciones realizadas por la aplicación.	- Ejecutar la aplicación en el teléfono - Seleccionar el botón atrás en el teléfono.
	EC5: El usuario recibe una llamada y vuelve a la aplicación	La aplicación debe pasar a un segundo plano y regresar luego de haber finalizado la llamada.	Ejecutar la aplicación en el teléfono - Introducir los datos en el formulario que muestra la aplicación. - Realizar llamada desde otro teléfono. - Colgar la llamada entrante.

Tabla 5: Caso de prueba Mostar alertas

Caso de prueba Mostrar alertas			
Nombre de Sección	**Escenario de Sección**	**Descripción de funcionalidad**	**Flujo Central**

NS2: Mostar alertas	ES1: El usuario tiene varias alertas en su perfil.	El sistema muestra en la interfaz las alertas correspondiente al usuario y las principales funcionalidades de la aplicación.	-Ejecutar la aplicación en el teléfono -Autenticarse satisfactoriamente
	ES2: El usuario no contiene alertas en su perfil.	El sistema muestra un mensaje informando que el usuario no contiene alertas	-Ejecutar la aplicación en el teléfono -Autenticarse Satisfactoriamente
	EC3: Ausencia de Conexión a Internet.	El sistema muestra un mensaje flotante informando que necesitar estar conectado a Internet para realizar la operación.	-Ejecutar la aplicación en el teléfono -Verificar la conexión -Autenticarse satisfactoriamente
	EC4: Seleccionar opción Atrás	El sistema debe finalizar todas las operaciones realizadas por la aplicación.	-Ejecutar la aplicación en el teléfono -Autenticarse Satisfactoriamente -Seleccionar el botón atrás en el teléfono
	EC5: El usuario recibe una llamada y vuelve a la aplicación	La aplicación debe pasar a un segundo plano y regresar luego de haber finalizado la llamada.	-Ejecutar la aplicación en el teléfono -Autenticarse Satisfactoriamente -Recibir llamada -Colgar la llamada entrante

Tabla 6: Descripción de Variables

Descripción de Variables: Autenticar Usuario				
NO	**Nombre del Campo**	**Clasificación**	**Valor Nulo**	**Descripción**
1	Usuario	Campo de texto	NO	Puede contener un nombre de usuario o una dirección de correo electrónico
2	Contraseña	Campo de Contraseña	NO	Puede contener todo tipo de caracteres
Descripción de Variables: Mostrar alertas				
NO	**Nombre del Campo**	**Clasificación**	**Valor Nulo**	**Descripción**
1	Usuario	Objeto de la Clase OrionUser	NO	Puede contener un nombre de usuario o una dirección de correo electrónico

3.5.2 Resultados de las pruebas de caja negra

Se efectuaron dos iteraciones, en la primera iteración según se muestra en la figura 9 se detectaron un total de ocho no conformidades. Luego de darle solución a las mismas se realizó una segunda iteración, en la que no se detectaron no conformidades.

Las no conformidades estuvieron asociadas a:

- Permisos de conexión en la aplicación Android.
- Dificultades en los códigos de estado de las respuestas del *APIRest*

- Parámetros innecesarios en las URIs para conexión con el *APIRest.*
- Gestión incorrecta de objetos en la aplicación Android.

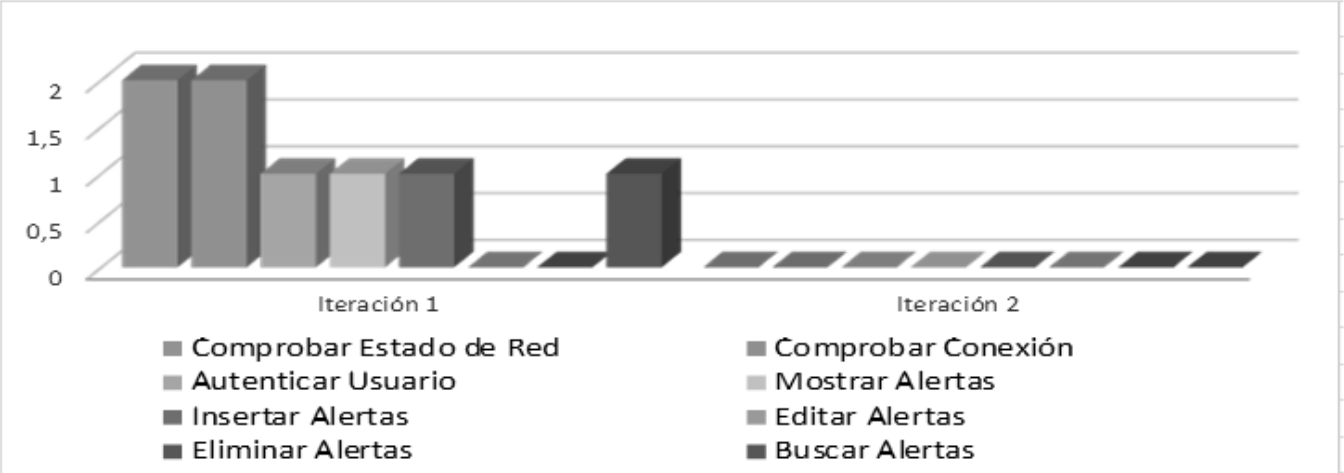


Ilustración 17: Pruebas de Caja negra

.

3.5.3 Pruebas de Rendimiento

De acuerdo a las características de la aplicación se decide realizar pruebas de rendimiento para comprobar el desempeño de la solución bajo distintos entornos de trabajo. Para llevar a cabo estas pruebas se utilizaron los siguientes dispositivos móviles:

- Samsung Galaxy S6. Versión de Android. CPU 8 Núcleos @2.1Ghz RAM: 3GB
- Samsung Galaxy J2. Versión de Android 5.2. CPU: 4 Núcleos @1.8 GHz RAM: 3GB.
- Acer Squile E600. Versión de Android: 4.2 CPU: 2 Núcleos @1.2GHz RAM: 2Gb.
- ALCATEL OT-4015x (POP C1). Versión Android 4.2.2 CPU: 2 Núcleos @1,0 GHz RAM:1GB

En la comparación se tuvo en cuenta el consumo de los siguientes recursos: tiempo de ejecución, batería, microprocesador y memoria RAM. A continuación, se muestran los resultados de las pruebas de rendimiento:

Tabla 7: Pruebas de Rendimiento

	Tiempo de Ejecución	Batería	Microprocesador	Memoria RAM
Samsung Galaxy S6	0.3s	5%	7%	12mb
Samsung Galaxy J2	0.5	4%	8%	12mb
Acer Squile E600	0.8	2%	13%	15mb
ALCATEL OT-4015x	0.9	5%	19%	14mb

Las pruebas de rendimiento se realizaron con el visor de recursos que contienen los dispositivos Android. Con la realización de esta prueba se pudo analizar el comportamiento de la aplicación en diferentes dispositivos móviles. Como se puede observar en la tabla, el consumo de recursos de la aplicación es mínimo en los cuatro escenarios. Lo cual es muy importante para usuarios con teléfonos de pocas prestaciones físicas o usuarios que tienen muchas aplicaciones ejecutadas en segundo plano y debido a esto no disponen de muchos recursos de *hardware.*

3.5.4 Pruebas de Integración

El proceso de integración del sistema implica construirlo a partir de sus componentes y probar el sistema resultante para encontrar problemas que pueden surgir debido a la integración. La integración del sistema implica identificar grupos de componentes que proporcionan alguna funcionalidad del sistema e integrar estos añadiendo código para hacer que funcione conjuntamente (Sommerville, 2005).

El motor de búsqueda cubano Orión, cuenta con una estructura que facilita la integración de nuevos subsistemas para incrementar sus funcionalidades. Por tal motivo, se decidió realizar pruebas de integración descendentes. Las cuales, consisten en desarrollar la infraestructura del sistema en su totalidad y luego añadirle los componentes funcionales (Sommerville, 2005).

Luego de desplegar el buscador Orión y comprobar que su funcionamiento es correcto se integró el *bundle APIRestBundle*. Durante este proceso obtuvieron los siguientes resultados

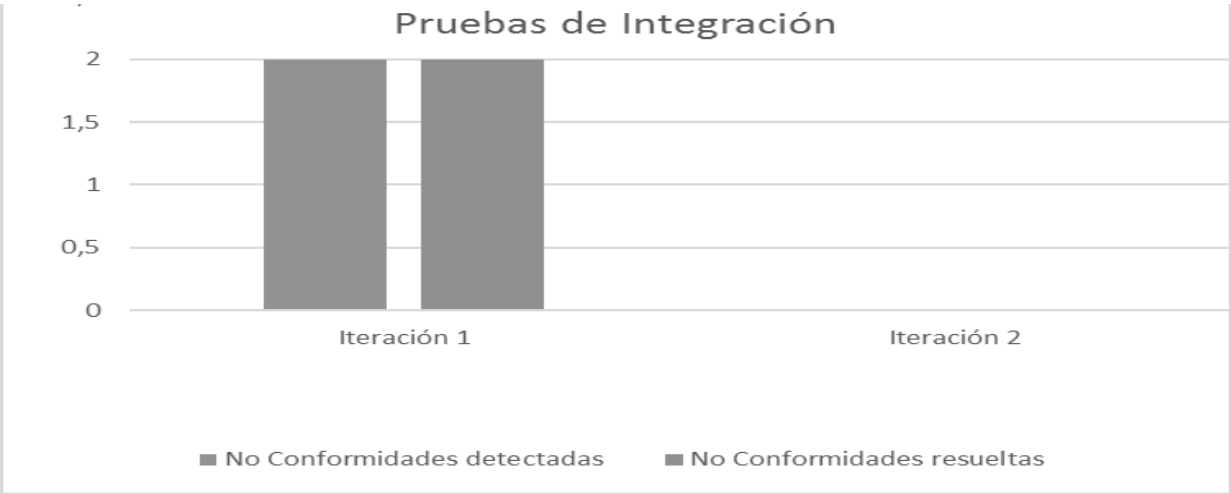


Ilustración 18: Pruebas de Integración

Las pruebas de integración realizadas permitieron detectar una dificultad en el proceso de creación de alertas en al APIRest*, específicamente en la asignación de usuario a las alertas generadas y una dificultad en la extracción de alertas de la base de datos. Dichas no conformidades fueron resueltas y realizada una segunda iteración en la que no se encontraron no conformidades. Luego de solucionar esta no conformidad, el* APIRest *desarrollado se integró correctamente con dicho motor de búsqueda.*

3.6 Conclusiones parciales

En el presente capítulo se han descrito los principales estándares de codificación, así como las principales nomenclaturas usadas en el código de la aplicación. Se ofrecen los resultados obtenidos en las pruebas realizadas tanto de caja negra como de rendimiento. Basándose en las actividades realizadas se arriba a las siguientes conclusiones:

- Los resultados de las pruebas de caja negra validan el cumplimiento de los requisitos funcionales definidos.
- Las nomenclaturas y estándares de codificación ayudan a realizar un código de programación más legible y profesional.
- Las pruebas de rendimiento muestran el comportamiento de la aplicación en un entorno real, lo cual es muy útil para comprobar el consumo de recursos físicos de la aplicación.

Conclusiones

- El análisis de sistemas homólogos permitió tener un acercamiento a las características y funcionalidades que debe tener la aplicación.
- La representación mediante *UML* permitió tener una visión más organizada para la etapa de implementación.
- El estudio de las tecnologías contribuyó a implementar una aplicación actualizada, así como la selección adecuada de las herramientas para la etapa de implementación.
- Se identificaron un total de 8 requisitos funcionales, siendo los requisitos de gestión de alertas los más relevante para el sistema.
- La realización de pruebas mostró el comportamiento del producto final en escenarios similares a los que será sometida la aplicación. Las cuales son muy útil para comprobar el consumo de recursos físicos de la aplicación y la validación de los requisitos funcionales.

Por todo lo expuesto anteriormente se concluye que los objetivos propuestos para el presente trabajo han sido cumplidos satisfactoriamente. La aplicación desarrollada contribuirá de manera significativa al sistema de alertas del buscador Orión.

Recomendaciones

A pesar de haberse cumplido los objetivos de este trabajo, en la elaboración del mismo surgieron algunas ideas para enriquecerlo de forma que sea una aplicación de mayores prestaciones:

- Realizar funcionalidades para la extracción de notificaciones de la base de datos
- Crear widgets para mostrar las notificaciones del usuario en la pantalla principal del dispositivo sin necesidad de ejecutar la aplicación directamente.

Referencia Bibliográfica

Aguilar, Ricardo.2016. Las mejores apps de noticias para Android. [En línea] 2016. [Citado el: 30/11/2016.] .http://andro4all.com/2016/02/mejores-apps-noticias-android-2016.

Android Developers. Get the Android SDK. [En línea].2014. [Citado el:2/12/2016.]. http://developer.android.com/sdk/index.htm.

LARMAN, C. UML y Patrones. Introducción al análisis y diseño orientado a objetos. s.l. : 1ra Edición, 1999. 970-17-0261-1.

Android, U. AndFTP cliente FTP para Android. 2009.[En línea] 2014 [Citado el:1/3/2017.]: http://www.universoandroid.com/2009/03/27/andftp-cliente-ftp-para-android/.

Caridad Orta Yudeikis Rodríguez, Pérez Guerrero Samuel. 2016. Módulo de traducción automática para el buscador Orión. [Citado el: 30/11/2016.].

Craig, Larman Prentice Hall. UML y Patrones. 2ª Edición. 2003. [Citado el: 21/11/2017.].

Cervantes, Humberto. Arquitectura de Software. [En línea] 2015. [Citado el: 23/02/2017]. https://sg.com.mx/revista/27/arquitectura-software.

Departamento de Ciencias de la Computacion e IA (DECSAI), Especificación de Requerimientos 2010. [En línea] 2014 [Citado el:14/3/2017.]. http://elvex.ugr.es/idbis/db/docs/design/2-requirements.pdf.

Davis, Williams. Herramientas CASE: Metodología estructurada para el desarrollo de los sistemas.1991. [Citado el:7/12/2016.].

Damaian, Calen. Eclipse ID. [En línea].2014[Citadoel:5/12/2016.]. http://www.genbetadev.com/herramientas/eclipse-ide.

Espeso, Pablo. 2015. Estadísticas de los navegadores web. [En línea] 2016. [Citado el: 15/10/2016.] http://www.xtro.es/statcounter-globalstats-estadisticas-de-navegacion/.

Fernandez Ruben. Patrones de diseño: qué son y por qué debes usarlos [En línea] 2014 [Citado el:1/3/2017.]. https://www.genbetadev.com/metodologias-de-programacion/patrones-de-diseno-que-son-y-por-que-debes-usarlos.

González. Sebastian Rincon [En línea]. 2012. [Citado el:6/4/2017.]: https://es.slideshare.net/rinconsete/pruebas-de-caja-blanca-y-negra.

González, M.2012. Cuba avanza hacia la migración del software libre. [En línea] 2012. [Citado el: 14/10/2016.]. http://www.somoslibres.org/modules.php?name=News&file=article&sid=2528.

González Nieto, Alejandro. ¿Qué es Android? [En línea] 2011. [Citado el:30/11/2016.] http://www.xatakandroid.com/sistema-operativo/que-es-android.

Granado, Ruiz Erick. Características de equipos móviles. [En línea] 2010 [Citado el:6/12/2016.] http://www.buenastareas.com/ensayos/Dispositivos-Moviles/800481.html.

González, Benjamín. XML:El lenguaje de los servicios web**.** 2004 [Citado el:7/12/2016.].

Gonzáles, Gornejo Jose Enrique. El Lenguaje de modelado Unificado. [En línea] 208. [Citado el: 7/12/2016]. http://www.docirs.com/uml.htm.

Ipaelo,2011. Android Market: Selección de las mejores aplicaciones por los editores (Editor´s Choice). [En línea] 2016. [Citado el: 30/11/2016.]. https://android.es/2011/05/23/android-market-seleccion-de-las-mejores-aplicaciones-por-los-editores-editors-choice/c.

Larramendi, Ferrás Beatriz Claudia. 2012.Propuesta de arquitectura para desarrollo de aplicaciones compuestas para dispositivos móviles con Android. [Citado el:2/12/2016.].

LARMAN, C. 2004. UML y Patrones:una introducción al analisis y diseño orientado a objetos y al proceso unificado. s.l. : Segunda.s.l.: Prentice hall.

Mena, David. 2016. Monitoriza tu presencia online con las alertas de Google. [En línea] 2016. [Citado el: 30/11/2016.] http://wanaleads.com/monitoriza-tu-presencia-online-con-las-alertas-de-google/.

Montano, José Luis Montes de Oca. 2015. SciELO Cuba. [En línea] octubre de 2015. [Citado el: 15 de 11 de 2016.] http://scielo.sld.cu/pdf/rus/v7n3/rus17315.pdf.

Munguía, Enrique. 2015. ¿Qué lenguaje de programación debo elegir? [En línea] 07 de diciembre de 2015. [Citado el: 08 de marzo de 2016.] http://www.enrique7mc.com/2015/12/que-lenguaje-deprogramacion-debo-elegir/.

Pastor, José Juan Castelló. 2016. Motores de búsqueda y derechos de autor: infracción y responsabilidad. s.l.:Aranzadi, 2016[Citado el: 14/10/2016.].

Página Oficial Rational. [En línea] 2007. [Citado el: 7/12/2016]. http://www.rational.com.ar/herramientas/roseenterprise.html.

Pressman, Roger S. Ingeneria de Software: Un enfoque práctico, 7ma edición, 2010.

Quintero, Jose. 2012. Eclipse como IDE para PHP. 2012. [Citado el:1/12/2016.].

Rosa, Juan Eladio Sanchez. Tuxpuc.Comparativas de framework en PHP:Cake PHP,Symfony y Zend Framework[En linea] 2 de septiembre 2013 [Citado el:1/12/2016.] http://tuxpuc.pucp.edu.pe/articulo/comparativa-de-framework-en-php-cakephp-sumfony-y-zend-framework.

República de Cuba. Partido Comunista de Cuba. (2011). Lineamientos de la política económica y social del Partido y la Revolución: La Habana: PCC. [Citado el: 16/10/2016.].

RODRIGUEZ CAMINO, Reinaldo. Motores de búsqueda sobre salud en Internet. ACIMED [En línea]. 2003, vol.11, n.5 [Citado 2016-10-20], pp. 0-0. http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1024-94352003000500002&lng=es&nrm=iso>.

Rumbaugh, James, Jacobson, Ivar y Booch, Grady. El Lenguaje Unificado de Modelado. Manual de Referencia. 2007. [Citado el: 7/12/2016].

Sommerville, Ian. 2005. Ingenieria de Software. Madrid : Pearson Eduación S.A , 2005. 84-7829-074-5.

Steven, Perry J. 2012. Introducción a la programación Java, parte 1: Conceptos básicos del lenguaje Java. 2012. [Citado el:30/11/2016.].

The PHP Group. ¿Qué es PHP?:PHP Manual. [En línea] 2011 [Citado el:1/12/2016.] http://www.php.net/manual/es/intro-whatis.php.

UCI . Metodología de desarrollo para la Actividad productiva de la UCI. Version 1.2 [Citado el: 28/02/ 2017.].

Visual Paradigm. Visual Paradigm for UML. [En línea] 2010. [Citado el: 7/12/2016]. http://www.visual-paradigm.com/.

Anexo I

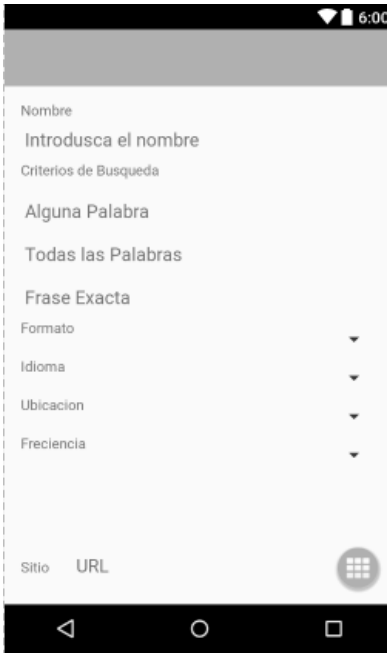
Tabla 8: Historia de usuario: Comprobar estado de red

Número:1	Nombre del requisito: Comprobar estado de red
Programador: Juan Pablo Díaz Ferrer	Iteración Asignada: 1
Prioridad: Alta	Puntos Estimados: 0.1
Riesgo en Desarrollo: Alto	Puntos Reales: 0.1
Descripción: El sistema debe comprobar si se encuentra conectado a la red para el correcto funcionamiento de la aplicación. Observaciones: Al inicia le aplicación debe mostrar un mensaje notificando si se encuentra conectado.	
Prototipo de interfaz: 	

Número:2	Nombre del requisito: Comprobar conexión	
Programador: Juan Pablo Díaz Ferrer	Iteración Asignada: 1	
Prioridad: Alta	Puntos Estimados: 0.1	
Riesgo en Desarrollo: Alto	Puntos Reales: 0.1	
Descripción: El sistema debe comprobar si se encuentra conectado con los servicios del *APIRest* implementado en el buscador Orión Observaciones: Al iniciar la aplicación debe mostrar un mensaje que notifique si se encuentra disponible los servicios del *APIRest* de Orión.		
Prototipo de interfaz: 		

Número:5	Nombre del requisito: Insertar alerta
Programador: Juan Pablo Díaz Ferrer	Iteración Asignada: 1
Prioridad: Alta	Puntos Estimados: 0.1
Riesgo en Desarrollo: Alto	Puntos Reales: 0.1
Descripción: El sistema debe permitir al usuario crear nuevas alertas. Observaciones :Al pulsar el botón de la interfaz debe mostrar un mensaje que notifique que la alerta ha sido insertada.	
Prototipo de interfaz: 	

Tabla 9:Historia de usuario: Editar Alerta

Número:6	Nombre del requisito: Editar alerta
Programador: Juan Pablo Díaz Ferrer	Iteración Asignada: 1
Prioridad: Alta	Puntos Estimados: 0.1
Riesgo en Desarrollo: Alto	Puntos Reales: 0.1
Descripción: El sistema debe permitir al usuario editar las alertas antes creadas. Observaciones: Se muestra la interfaz del requisito funcional Insertar Alerta, pero con los campos rellenados por los datos de la alerta que se desea modificar. Al pulsar en el botón de la interfaz debe mostrar un mensaje notificando que la alerta ha sido modificada.	
Prototipo de interfaz: 	

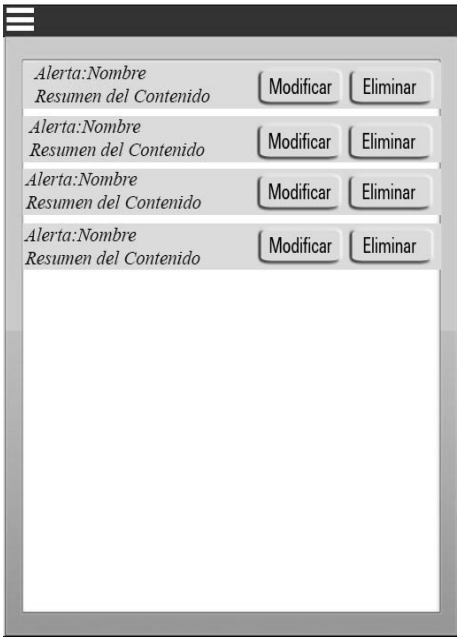
Número:7	Nombre del requisito: Eliminar alerta
Programador: Juan Pablo Díaz Ferrer	Iteración Asignada: 1
Prioridad: Alta	Puntos Estimados: 0.1
Riesgo en Desarrollo: Alto	Puntos Reales: 0.1
Descripción: El sistema debe permitir al usuario eliminar alertas. Observaciones: Al pulsar el botón eliminar de una de las alertas debe mostrar un cuadro de dialogo para tener seguridad que el usuario desea eliminar la alerta. Al pulsar el botón de aceptar en dicho cuadro de dialogo se debe eliminar la alerta del sistema y mostrar una notificación de alerta eliminada.	
Prototipo de interfaz:  	

Tabla 10: Historia de usuario: Buscar Alertas

Número:8	Nombre del requisito: Buscar alertas
Programador: Juan Pablo Díaz Ferrer	Iteración Asignada: 1
Prioridad: Alta	Puntos Estimados: 0.1
Riesgo en Desarrollo: Alto	Puntos Reales: 0.1
Descripción El sistema debe permitir al usuario Buscar Alertas. Observaciones: Al pulsar el botón se debe mostrar la alertas que contenga en el campo nombre las palabras introducidas por el usuario. En caso de no existir ninguna coincidencia se muestra un mensaje notificando la ausencia de alertas con ese criterio de búsqueda.	
Prototipo de interfaz: