Facultad 5

Trabajo de diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Título: Propuesta de arquitectura para desarrollo de aplicaciones compuestas para dispositivos móviles con Android.

Autor: Claudia Beatriz Larramendi Ferrás

Tutor: Ing. Orestes Febles Díaz

La Habana, 2012

DECLARACIÓN DE AUTORÍA

Por este medio se declara que soy la única autora de este trabajo y se autoriza a la Universidad de las Ciencias Informáticas (UCI) para que haga el uso que estime pertinente con este trabajo.

Para que así conste se firma la presente a los __ días del mes de _____ del 2012.

_____	_____
Firma de la Autora	Firma del Tutor
(Claudia Beatriz Larramendi Ferrás)	(Ing. Orestes Febles Díaz)

DATOS DE CONTACTO

Ing. Orestes Febles Díaz (ofebles@uci.cu)

Graduado con el título de Ingeniero en Ciencias Informáticas en el 2008, en la Universidad de las Ciencias Informáticas.

Dedicatoria

Mi esfuerzo durante estos cinco años de la carrera, todos mis resultados como estudiante, mis logros personales y mi título de Ingeniera en Ciencias Informáticas, van dedicados con todo cariño a mis abuelos Ángel y Damaris, y a mi tío Angelito; que aunque ya no están conmigo físicamente siempre van en mí corazón.

Agradecimientos

A mi madre, la persona que más quiero y admiro, gracias por quererme y apoyarme en todas mis decisiones, gracias por la dedicación y el cariño que siempre me has dado.

A mi papá Machy, por estar presente todos estos años y tratarme como una hija.

A mi padre, por su forma de ser que me ha ayudado a ser más fuerte y distinta a los demás.

A mi tía Isa, por ayudarme con la segunda inteligencia, por sus consejos que más de una vez me han sacado del apuro.

A mi hermano Octavito, por ser la razón de muchos de mis logros, al querer ser un ejemplo para él.

A mi hermano Davel, por servirme de inspiración para coger esta carrera.

A toda mi familia, por estar siempre pendientes de mí.

A Julio, por demostrarme que es posible tenerlo todo en una sola persona, por ser mi compañero incondicional, mi novio y mi amigo más especial.

A mi segunda familia Leanys, Yadira, Daniel y Yadiel, por estar a mi lado en los momentos buenos y en los malos.

A Fabio, por ayudarme en toda la carrera desde primer año con matemática, hasta quinto con la tesis.

A mis amistades Lauren, Ernesto, Noly, Anacelia, Hecty, Lito, el piquete “Motos” y el grupo 5506.

A mis campeones del ludus, por el respeto y la paciencia que tuvieron siempre que me permitieron jugar futbol con ustedes

A mi tutor Orestes y los profesores Jessie, Luis Enrique, Rubén y Aymé, que colaboraron con entrega y paciencia a la realización de esta tesis.

A todas las personas con las que he compartido durante estos cinco años, gracias por hacer de la UCI un lugar único que nunca olvidaré.

Resumen

El uso de una arquitectura de software orientada a servicios mejora el proceso de desarrollo de software, aportando agilidad durante la construcción del producto y calidad al resultado. Además promueve el desarrollo de aplicaciones compuestas, formadas por servicios; permitiendo la reutilización de componentes, y por consiguiente la racionalización de la cantidad de aplicaciones necesarias para resolver los problemas de determinada organización.

Este trabajo propone una arquitectura para el desarrollo de aplicaciones compuestas para dispositivos móviles equipados con el sistema operativo Android. El uso de esta arquitectura permitirá a los desarrolladores crear soluciones informáticas que utilicen recursos propios o externos, tales como bases de datos, servicios y páginas web. Se definen patrones de diseño, herramientas y frameworks que constituyen una guía muy completa para mejorar considerablemente la construcción de sistemas informáticos sobre dicha plataforma móvil.

La propuesta arquitectónica fue probada mediante la implementación de un caso de estudio práctico sobre un escenario real en la Universidad de las Ciencias Informáticas; demostrando la factibilidad de los componentes propuestos.

Palabras clave: arquitectura, aplicaciones, Android.

ÍNDICE

Índice de Figuras

Índice de Tablas

Introducción

Las tecnologías inalámbricas han mantenido un auge creciente en estos últimos años, entre ellas ha tenido un desarrollo favorable la telefonía móvil, que desde sus inicios a finales de la década del 70 ha revolucionado enormemente las actividades que se realizan a diario. Los teléfonos celulares se han convertido en una herramienta primordial tanto para las personas comunes como para los agentes de negocios.

En el momento de su creación la tecnología móvil fue concebida únicamente para manejar la voz del ser humano, sin embrago los avances tecnológicos actuales han extendido su uso y hoy día es capaz de brindar servicios tales como el envío y almacenamiento de archivos de audio y video.

El avance de las Tecnologías de la Información y las Comunicaciones (TIC) ha sido constante como punto convergente y de integración entre la computación, la microelectrónica y las telecomunicaciones, esto las convierte en el conjunto de medios (la radio, la televisión y la telefonía) que optimizan la comunicación y que permiten la captura, reproducción, tratamiento y presentación de informaciones; en forma de voz, imágenes y datos contenidos en señales de naturaleza acústica, óptica o electromagnética.

Dentro de las TIC se encuentran los llamados dispositivos móviles, aparatos de pequeño formato, los cuales también han reconocido varias generaciones evolutivas y cambios trascendentales.

Según los datos reportados por la firma consultora Gartner, en el 2011 a nivel mundial las ventas de dispositivos móviles alcanzaron un total de 476.5 millones de unidades en el cuarto trimestre del año, lo que representa un 5.4 % de incremento sobre el mismo periodo del 2010. De manera acumulada en el año, los consumidores compraron mil 800 millones de unidades, esto constituye un 11.1 % de crecimiento total sobre el 2010.[1]

Un dispositivo móvil debe ser lo suficientemente pequeño para ser transportado y utilizado durante su transportación, además debe tener capacidad de procesamiento, almacenaje limitado y conexión a redes por diferentes vías. En el mundo de los dispositivos móviles se incluyen reproductores de audio, ordenadores portátiles, cámaras digitales, agendas electrónicas, localizadores, buscapersonas, beepers, teléfonos inteligentes y ordenadores personales en tableta; los cuales normalmente se equipan con un sistema operativo para manejar, controlar y actualizar sus aplicaciones y datos.

La gama de dispositivos móviles creados durante los últimos años es muy amplia; entre los sistemas operativos disponibles para equiparlos se encuentran: Linux, BlackBerry OS

(BlackBerry Operating System o Sistema Operativo BlackBerry), Windows Mobile, iPhone OS y Android.

Los sistemas operativos móviles son más simples que los empleados en las computadoras, están más orientados a la conectividad inalámbrica, los formatos multimedia para móviles y las diferentes maneras de introducir información en ellos.

Android es la plataforma creada por Google. Se ha convertido en el sistema operativo móvil más utilizado, alcanzando la cifra de 850.000 dispositivos activados diariamente.[2]

Hoy en día existen alrededor de 450.000 aplicaciones disponibles en la tienda oficial de Android: Android Market, estas pueden ser gratuitas o adquirirse mediante un pago. Su clasificación por tipo se divide en: compras, comunicación, deportes, finanzas, herramientas, multimedia, noticias y meteorología, ocio, productividad, salud, sociedad, viajes, educación y bibliotecas de software.[3]

Las aplicaciones para dispositivos móviles pueden ser simples o compuestas. Mientras una aplicación simple debe contener dentro de sí misma todos los componentes que le permitan cumplir el objetivo para el que fue creada, una aplicación compuesta tendrá la posibilidad de utilizar e integrar recursos como bases de datos y servicios publicados o suministrados por otras aplicaciones.

En Cuba el uso masivo de la tecnología móvil ha crecido considerablemente durante últimos años, debido al aumento de las ventas de líneas y teléfonos celulares a toda la población.

La Empresa de Telecomunicaciones de Cuba S.A (ETECSA) es la encargada de comercializar los accesorios y servicios de telecomunicaciones en el país. Esta cuenta con entidades colaboradoras como la Universidad de las Ciencias Informáticas (UCI) y DESOFT para el desarrollo de aplicaciones para dispositivos móviles. Este desarrollo en muchos casos se ha visto afectado por las consecuencias del bloqueo económico, que se evidencian en la denegación del acceso a servicios informáticos tales como sitios web y foros tecnológicos, además en la difícil adquisición de algunas tecnologías modernas. Esto ha motivado que el país esté inmerso en el proceso de alcanzar la soberanía tecnológica mediante la migración a software libre, ya que su utilización no implica gastos adicionales por concepto de cambio de plataforma ni la adquisición de licencias de software privativos.

El Centro de Consultoría y Desarrollo de Arquitecturas Empresariales (CDAE), se enmarca dentro de la estrategia de la UCI de creación de centros de desarrollo, con el objetivo de elevar el nivel de respuesta frente a las crecientes demandas para la informatización del país y de exportación de productos informáticos y servicios profesionales; dentro de las

temáticas que aborda se encuentra el desarrollo de aplicaciones compuestas en el contexto de una Arquitectura Orientada a Servicios (SOA, del inglés Service Oriented Architecture).

El desarrollo de aplicaciones compuestas ha significado un cambio de paradigma. Componer una aplicación a partir de funcionalidades existentes que pueden ser reutilizadas conlleva a un menor gasto de recursos humanos, materiales y de tiempo de desarrollo, en su mayoría los mismos componentes podrían soportar determinada funcionalidad del negocio en diferentes sistemas. Estos conceptos y bondades también pueden ser aplicados al desarrollo de aplicaciones compuestas para dispositivos móviles que utilicen Android como sistema operativo.

Actualmente los desarrolladores de la UCI que incursionan en el trabajo con la plataforma Android se enfrentan a un proceso dificultado, con demoras innecesarias y afectaciones en la calidad del resultado final.

De la situación problémica antes expuesta se deriva el siguiente **problema de investigación** a resolver: ¿Cómo mejorar el desarrollo de aplicaciones compuestas para dispositivos móviles que utilicen el sistema operativo Android en el entorno tecnológico de la UCI?

Luego, se identifica como **objeto de estudio:** las arquitecturas para dispositivos móviles y como **campo de acción** las arquitecturas para dispositivos móviles con Android.

Para darle solución al problema de investigación se propone como **objetivo general de la investigación** elaborar una propuesta arquitectónica factible para el desarrollo de aplicaciones compuestas para dispositivos móviles que utilicen Android.

Este objetivo general puede desglosarse en los siguientes **objetivos específicos**:

- Realizar el estudio del estado del arte sobre el sistema operativo Android, las aplicaciones compuestas y el diseño de arquitecturas de desarrollo.
- Realizar una propuesta arquitectónica de desarrollo de aplicaciones compuestas con Android.
- Completar un caso de estudio real para aplicar la propuesta arquitectónica presentada y validar mediante el criterio de expertos.

La **idea a defender** es que con la investigación se obtendrá, una propuesta arquitectónica con todos sus elementos de arquitectura lógica de la solución, que mejorará considerablemente el proceso de desarrollo de aplicaciones compuestas para dispositivos móviles con el sistema operativo Android.

Diseño metodológico de la investigación:

La **estrategia de investigación** utilizada:

Investigación exploratoria: Para conocer la situación de las arquitecturas de software orientadas a servicios y de Android.

Los métodos utilizados en la investigación:

Métodos teóricos:

- **Analítico – Sintético:** Para analizar el sistema operativo Android y las aplicaciones compuestas.
- **Inductivo – Deductivo:** Esta investigación propicia determinar cuáles son los aspectos necesarios para definir una propuesta arquitectónica.

Métodos empíricos:

- **Método encuesta:** Para realizar consultas a las personas que abordan las temáticas de arquitecturas de software, aplicaciones compuestas y Android.

La estructura del presente trabajo de diploma consta de tres capítulos, donde se expone lo referente a las aplicaciones compuestas para Android, el diseño de una arquitectura de desarrollo para estas y un estudio detallado sobre la implementación de un ejemplo práctico que demuestra la validez de la propuesta arquitectónica. Para cada capítulo se realizan la introducción y las conclusiones parciales.

En el primer capítulo se ofrece un estudio del estado del arte, donde se exponen algunos conceptos relacionados con la tecnología Android y las aplicaciones compuestas.

En el segundo capítulo se expone detalladamente la propuesta arquitectónica desarrollada.

En el tercer capítulo se brinda la solución implementada para un caso de estudio práctico sobre un escenario real en la Universidad de las Ciencias Informáticas, que evalúa y valida la propuesta arquitectónica.

Capítulo 1 Fundamentación teórica

1.1 Introducción

En este capítulo se abordan como temas centrales las arquitecturas de software para dispositivos móviles, las aplicaciones compuestas y el sistema operativo Android. Se presentan conceptos esenciales, arquitecturas de referencia, características de las aplicaciones compuestas y el estado del arte de la plataforma de desarrollo de Android.

1.2 Fundamentos de las arquitecturas de software

Algunos conceptos de ingeniería de software plantean:

- La arquitectura de software de un programa o de un sistema computacional está definida por la estructura, comprendida por los elementos de software, las propiedades visibles de esos elementos y las relaciones entre ellos.[4]
- La arquitectura de software es, a grandes rasgos, una vista del sistema que incluye los componentes principales del mismo, la conducta de esos componentes según se la percibe desde el resto del sistema y las formas en que los componentes interactúan y se coordinan para alcanzar la misión del sistema.[5]
- Arquitectura es la organización fundamental de un sistema descrita en: sus componentes, la relación entre ellos y con el ambiente, los principios que guían su diseño y evolución. Es la estructura de un sistema, la cual abarca componentes de software, propiedades externas visibles de estos componentes y sus relaciones.[6]

Debe entenderse entonces que la arquitectura de software es la base para representar la estructura de un sistema de una manera abstracta y fácil de interpretar. Una buena arquitectura explica claramente los elementos que la forman y la manera en que se comunican entre ellos.

El proceso de desarrollo de software es aquel en que las necesidades del usuario son traducidas en requerimientos de software, estos requerimientos transformados en diseño y el diseño implementado en código; el código es probado, documentado y certificado para su uso operativo.

La arquitectura de software constituye una guía para realizar el proceso de desarrollo de software. Si durante todo el ciclo de vida de un software, el equipo de trabajo se rige por una arquitectura previamente definida según los requisitos del cliente, el proceso de desarrollo será agilizado y optimizado cualitativamente.

1.3 Elementos que definen una arquitectura de software para dispositivos móviles

Los arquitectos de soluciones para dispositivos móviles deben considerar una cantidad de factores limitantes, muchos más que al generar aplicaciones para una computadora personal. Algunos desafíos que se enfrentan al construir arquitecturas en el entorno móvil son [7]:

1. Los dispositivos tienen recursos limitados. A medida que los dispositivos y sus componentes pasan a ser más pequeños y livianos, la energía disponible para respaldar aplicaciones ricas disminuye. También, la capacidad de la memoria del dispositivo es limitada, aunque esto se mitiga mediante mejoras en la tecnología de almacenamiento.
2. Los dispositivos no están estandarizados. Los arquitectos de soluciones móviles deberán optimizar el software y el hardware para obtener experiencias más completas. La falta de estandarización del hardware obliga a los desarrolladores a suponer el grupo de recursos que estarán disponibles para la aplicación.
3. Los dispositivos deben soportar escenarios sin conexión y a veces, conexiones lentas. Necesitan ser más que pantallas de cliente liviano; deben ser plataformas de aplicación por sí mismos. Un requisito clave para esta capacidad es el almacenamiento y procesamiento local, con sincronización con ordenadores o servicios de Internet.
4. La conectividad no está estandarizada. En función de las capacidades de los dispositivos y del plan de servicios que tiene con su operador de red, un usuario puede acceder a información y servicios en Internet móvil a través de la voz, mensajería (mensajes cortos, correo electrónico) o a través de protocolos de Internet (Wi-Fi, conexión relacionada o plan de datos adecuado). En los mercados emergentes es probable que los servicios deban ofrecerse sobre SMS (Short Message Service o Servicio de Mensajes Cortos) o voz ya que es más usual que los usuarios posean teléfonos masivos con capacidades muy limitadas. Para experiencias ricas de multimedia se necesita un protocolo de Internet rápido.

1.3.1 Formato de mensajería

El formato de mensajería para dispositivos móviles define la longitud máxima y el contenido que podemos incorporar al mensaje; en algunos casos es solo texto, en otros también es posible enviar archivos multimedia como animaciones, imágenes, sonidos y videos.

Actualmente se dispone de varios formatos, entre ellos [8]:

- SMS: Es una secuencia de hasta 140 caracteres o de 160 caracteres de 7 bits, que incluye parámetros como la fecha de envío, el número telefónico del destinatario y del remitente, y el código del centro de servicio de mensajes cortos SMSC (Short Message Service Center). Son enviados a través de la red estándar de telefonía digital GSM (Global System for Mobile communications o Sistema Global para las comunicaciones Móviles); debido a su pequeño tamaño pueden ser enviados y recibidos en cualquier momento, incluso durante una llamada.
- MMS (Multimedia Messaging System o Sistema de Mensajería Multimedia): Es una extensión de la mensajería móvil que permite incorporar contenido multimedia a los mensajes. La longitud válida no es limitada por este estándar, sino por los operadores del sistema GSM, al que se incorpora un MMSC (Multimedia Message Service Center o Centro de Mensajes Multimedia) encargado de gestionar los mensajes. Para sincronizar los contenidos se utiliza el protocolo SMIL (Synchronized Multimedia Integration Language o Lenguaje Sincronizado Multimedia)

1.3.2 Marcos de trabajo

Un marco de trabajo en el desarrollo de software es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto. [9]

Otra definición expresa que un marco de trabajo (también conocido como framework) es una solución completa que incluye herramientas de apoyo a la fabricación de aplicaciones (ambiente de trabajo o desarrollo) y motores de ejecución (ambiente de ejecución). Un framework puede ser algo tan grande como Java, pero también el concepto se aplica a ámbitos más específicos, por ejemplo: dentro de Java en el ámbito específico de aplicaciones Web se tienen los frameworks: Struts, Java Server Faces y Spring. Estos frameworks de Java en la práctica son conjuntos de librerías para desarrollar aplicaciones Web, además librerías para su ejecución y un conjunto de herramientas para facilitar esta tarea.[10]

1.3.3 Patrones de diseño

Los patrones de diseño han sido propuestos en varios escenarios como una vía para comunicar de forma compacta el problema a resolver, capturando su esencia y la de su solución; sus componentes son las clases y objetos, así como los mecanismos de interacción son los mensajes. El conjunto de patrones para un dominio específico, con el

tiempo forma un lenguaje de patrones que luego es reutilizado durante el proceso desarrollo. Para el caso de las aplicaciones móviles, el hecho de disponer de un lenguaje de patrones, tiene una serie de beneficios potenciales, por ejemplo se tendrá una forma estándar de documentar las lecciones aprendidas en los primeros diseños. Algunos patrones de diseño son: Abstract Factory, Adapter, Bridge, Builder, Chain of Responsibility, Command, Composite, Decorator, Facade, Factory Method, Flyweight, Interpreter, Iterator, Mediator, Memento, Observer, Prototype, Proxy, Singleton, State, Strategy, Template Method y Visitor [11]

1.3.4 Patrones de arquitectura

El concepto de patrones arquitectónicos surgió algunos años atrás y es similar a la noción de patrones de diseño, pero los patrones arquitectónicos tratan con niveles de abstracción muy superiores, donde los componentes ya no son clases y objetos sino por ejemplo módulos de software y las interacciones entre estos componentes se realizan a través de conectores. En las aplicaciones móviles los patrones arquitectónicos son clasificados como de: acceso, adaptación, personalización e interfaz. Algunos patrones de arquitectura son: Layers, Pipes & Filters, Blackboard, Broker, Model-View-Controller, Presentation-Abstraction-Control, Microkernel y Reflection. [11]

1.3.5 Plataforma de sistema operativo

Existen numerosas plataformas de sistemas operativos para el desarrollo de aplicaciones móviles, varían en cuanto al lenguaje de programación que utilizan, las herramientas disponibles, la forma de generar los gráficos y presentarlos en la interfaz de usuario. Cada uno de estos sistemas cuenta con un mercado de aplicaciones independiente donde algunas son gratis y otras no. Android es una de estas plataformas, que rompió paradigmas al ser liberado su código de fuente; existen además BlackBerry OS, iOS, Palm OS, Windows Phone, entre otros.

1.3.6 Portabilidad

Los usuarios actualmente emplean los teléfonos móviles como una cámara, una oficina móvil, una consola de juegos, un asistente personal y mucho más, gracias a la gran cantidad de aplicaciones que están disponibles para servir a propósitos innumerables. La portabilidad en las aplicaciones y sistemas operativos para dispositivos móviles es un aspecto de alta prioridad. Las capacidades de almacenamiento limitadas que caracterizan al entorno móvil representan un reto significativo para los desarrolladores que deben cumplir con las expectativas de los clientes en cuanto a poder llevar la información y utilizar las

aplicaciones deseadas en cualquier lugar y momento. Aunque el tamaño de memoria de los dispositivos se puede expandir mediante tarjetas físicas, un sistema óptimo para estos medios no debe ocupar más que unos pocos kilobytes.

1.3.7 Protocolo de comunicación con servidores

Los programas servidores normalmente reciben las solicitudes de los programas clientes, ejecutan algún tipo de manipulación con los datos (recuperar datos de una base de datos, efectuar algún cálculo, implementar alguna regla de negocio, etc.), y envían la respuesta apropiada al cliente.

Un protocolo de comunicación Cliente/Servidor establece las reglas para el intercambio de datos y servicios entre estas entidades. Los dispositivos móviles emplean protocolos inalámbricos como WAP (Wireless Application Protocol); también el protocolo de red TCP/IP (Transmission Control Protocol/Internet Protocol o Protocolo de Control de Transmisión/Protocolo de Internet) que presta servicios de FTP (File Transfer Protocol) para la transferencia de archivos en la red, de acceso remoto a otros ordenadores, de correo electrónico, de NFS (Network File System) para obtener archivos de otros computadores; y HTTP (Hypertext Transfer Protocol) para comunicarse con los navegadores.[12]

1.3.8 Rendimiento

El rendimiento de las aplicaciones móviles puede verse afectado por un ancho de banda insuficiente, costoso o congestionado en las oficinas remotas; o la falta de visibilidad con los servidores de la red; o por un desbalance en cuanto al consumo de energía. Algunos sistemas permiten la ejecución simultánea de varias aplicaciones, estableciendo patrones de prioridad sobre ellas y manteniendo solo una en el primer plano del dispositivo. Lo ideal es mejorar el rendimiento del dispositivo sin afectar al usuario.

1.3.9 Seguridad

La seguridad es una de las mayores preocupaciones para los entornos móviles. Las soluciones más robustas sobre este aspecto deben abarcar todo el trayecto móvil, desde los sistemas de “back-end” (puertas-traseras) hasta el dispositivo del cliente. Puede decirse que la seguridad en una aplicación para dispositivos móviles tiene tres aspectos principales [13]:

- Privacidad de la mensajería: Asegurar que los datos sensibles no estén comprometidos durante su transmisión. Esto comúnmente se realiza con el uso de criptografía.

- Autentificación: Asegurar que la identidad de todos los usuarios involucrados en la comunicación es correcta. Usualmente efectuado con el empleo de certificados digitales o técnicas de usuario/contraseña.
- Seguridad en el dispositivo: Asegurar la protección de los datos almacenados en el dispositivo móvil, usualmente logrado a través del uso de una contraseña de protección y encriptación.

Según la arquitectura, se dispone de distintos mecanismos de seguridad ya provistos por sistemas comerciales, que se pueden adherir a la aplicación móvil.

1.3.10 Usabilidad

La usabilidad para los dispositivos móviles está fuertemente relacionada con 4 aspectos importantes [11]:

- ✓ Disponibilidad: Los dispositivos móviles y sus aplicaciones deben mantenerse activos 24 horas al día, 7 días a la semana. Los teléfonos celulares y PDAs (Personal Digital Assistant) son un ejemplo de esto, los mismos siempre se encuentran activos o en algún modo de espera que les asegura estar disponibles casi instantáneamente cuando se los requiera.
- ✓ Manejo de fallas: Las aplicaciones ejecutadas desde un dispositivo móvil, operan en un entorno de alta demanda, donde ocurren problemas como las interrupciones en la comunicación, la disminución del ancho de banda, la capacidad limitada de la batería, entre otros. Este tipo de aplicación debe garantizar que los datos importantes para el usuario y la administración del dispositivo puedan superar la ocurrencia de daños inesperados o fallas.
- ✓ Uso de la memoria: En los dispositivos móviles comúnmente no se cuenta con grandes discos de almacenamiento cuya memoria pueda ser aprovechada por las aplicaciones, en caso de que no esté siendo usada por algún archivo. Debido a esto, para mantener un máximo rendimiento, se mantiene todo en la memoria física, donde el acceso es mucho más veloz.
- ✓ Servicios Críticos: Varios dispositivos móviles sirven a otro propósito crítico mientras se ejecutan aplicaciones prioritarias. La mayoría de los sistemas operativos móviles poseen niveles de protección para las funciones críticas, pero de todas formas una aplicación móvil debe ser capaz de hacer que el dispositivo siga siendo útil bajo cualquier circunstancia, asegurando que una serie de servicios críticos estén disponibles para el usuario en todo momento.

1.4 Arquitecturas de referencia para dispositivos móviles

RUP (Rational Unified Process o Proceso Unificado de Rational) plantea que una arquitectura de referencia es un patrón, o conjunto de patrones predefinidos, con posibilidad de ser instanciados completa o parcialmente, diseñados y probados para su uso en un proyecto y contexto tecnológicos particulares, en conjunto con los artefactos de soporte para posibilitar su empleo. [14]

Una arquitectura de referencia puede contribuir en los siguientes aspectos: [15]

- Reducir el tiempo de preparación de un proyecto.
- Reducir el tiempo y costo del desarrollo de aplicaciones.
- Incrementar la calidad del producto.
- Asegurar operaciones propias de las aplicaciones.
- Reducir el costo del mantenimiento.

En el entorno de desarrollo de aplicaciones para dispositivos móviles se puede decir que la utilización de una arquitectura de referencia proporciona un marco para la toma de decisiones en función de reducir el riesgo técnico de los proyectos individuales, además de definir un conjunto de directrices arquitectónicas a compartir entre sistemas y aplicaciones (plataforma y herramientas de desarrollo, patrones de diseño, componentes para mecanismos de seguridad y acceso a datos, etc.).

1.4.1 Arquitectura Java 2 Micro Edition

Más conocida como J2ME, es la plataforma basada en el lenguaje Java que Sun Microsystems ha creado para la programación de dispositivos inalámbricos pequeños como teléfonos celulares, localizadores y PDAs.

Al igual que sucede con J2EE (Java 2 Enterprise Edition), que está orientado a entornos corporativos o J2SE (Java 2 Standard Edition), orientado a sistemas de sobremesa, la arquitectura J2ME está formada por un conjunto de APIs (Application Programming Interface o Interfaz de programación de aplicaciones) estándares que permiten que las aplicaciones desarrolladas bajo esta arquitectura se beneficien de las características multiplataforma de Java y que abren la puerta a la distribución de aplicaciones a millones de dispositivos.

Es utilizado por diversas compañías como son Alcatel, BlackBerry, Casio, Fujitsu, Hitachi, Kyocera, LG Electronics, Mitsubishi, Motorola, NEC, Nokia, Panasonic, Samsung, Sanyo, Sharp, Siemens, Sony Ericsson, Toshiba.

Esta edición micro de Java se compone, además del lenguaje, de una máquina virtual, configuraciones, perfiles y paquetes adicionales, que se muestran en la siguiente figura:

Figura 1 - Arquitectura de J2ME

Esta arquitectura presenta como frameworks para interfaz: RMS (Record Management System), para la persistencia de datos: Floggy y para el desarrollo OpenBaseMovil, Solid-Mobile, Mojax.

Las opciones de gestor de base de datos para el lenguaje SQL en esta arquitectura son: PointBase Micro, SimpleOODBMS y Perst LiteBertiente. Utiliza los protocolos HTTP, HTTPS (Hypertext Transfer Protocol Secure), TCP/IP, UDP (User Datagram Protocol) y WAP. J2ME puede ser usado por los sistemas operativos Linux, Solaris y Windows.

Existen varios entornos de desarrollo de diferentes proveedores: Sun Java Studio Mobility 6 de Sun Microsystems, JBuilder X Mobile Edition de Borland, NetBeans Mobility Pack 4.0 de Netbeans.org y EclipseME J2ME Development de Sourceforge.net.

Entre las herramientas de desarrollo de aplicaciones están: J2ME Wireless Toolkit, Nokia Developer's Suite, Siemens Mobility Toolkit, Motorola iDEN, Samsung Java SDK (Software Development Kit o kit de desarrollo de software) y SonyEricsson J2ME SDKs. [16]

1.4.2 Arquitectura de Android

Google adquirió la compañía Android Inc. y desarrolló la plataforma Android, posteriormente en acuerdo con el consorcio Open Handset Alliance decide promocionar los estándares de códigos abiertos para dispositivos móviles.

Entre las características notables se tiene 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); 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 (Global Positioning System o Sistema de Posicionamiento Global), acelerómetros, giroscopios, magnetómetros, termómetros, sensores de proximidad y de presión). La siguiente imagen muestra los componentes principales de la arquitectura Android:

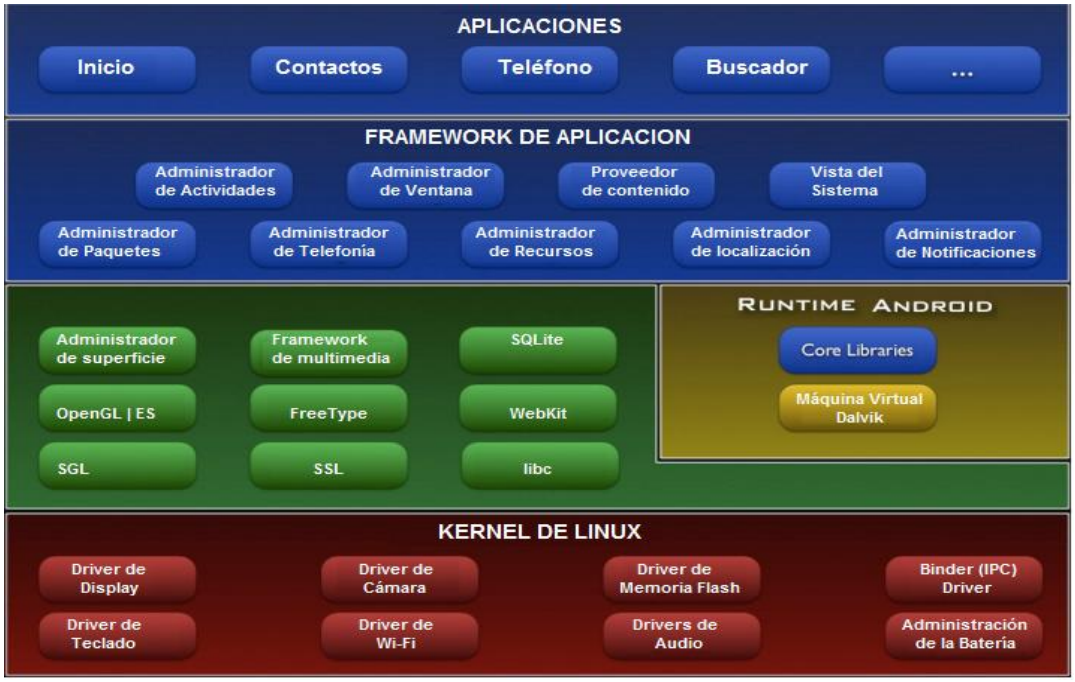

Figura 2 - Arquitectura de Android OS

Para realizar aplicaciones sobre la plataforma Android se instala el plugin ADT y el SDK en el IDE (Integrated Development Environment o Entorno de Desarrollo Integrado) de desarrollo. Un SDK de Android contiene las librerías, el emulador, documentación, ejemplos de código, tutoriales y una versión de la plataforma. Se usa el lenguaje de programación Java. [17]

Actualmente existen seis fabricantes: Dell, HTC, Kyocera, LG, Motorola, y Samsung, produciendo teléfonos inteligentes que usan el sistema operativo Android.

1.4.3 Arquitectura de BlackBerry OS

La compañía canadiense RIM (Research In Motion) desarrolló el sistema de los teléfonos inteligente BlackBerry. El sistema operativo de BlackBerry opera con subprocesos múltiples,

esto significa que numerosas aplicaciones y procesos se pueden ejecutar de forma activa en el dispositivo BlackBerry al mismo tiempo.

Las aplicaciones principales de BlackBerry (incluida la lista de mensajes, la lista de contactos, el calendario y el explorador) se crean como aplicaciones con configuración CLDC, esto significa que no se pueden trasladar a otros dispositivos

Los dispositivos BlackBerry proporcionan un entorno inalámbrico JavaME compatible con las aplicaciones cliente/servidor. Las aplicaciones pueden comunicarse con las redes mediante conexiones estándar TCP/IP y HTTP, además de usar la red inalámbrica subyacente con protocolo WAP.

El BlackBerry MDS (BlackBerry Mobile Data System) proporciona proxies HTTP y TCP/IP que permiten al dispositivo BlackBerry comunicar la aplicación con los servidores tras el firewall de la empresa sin software adicional, como se puede apreciar en la siguiente imagen:

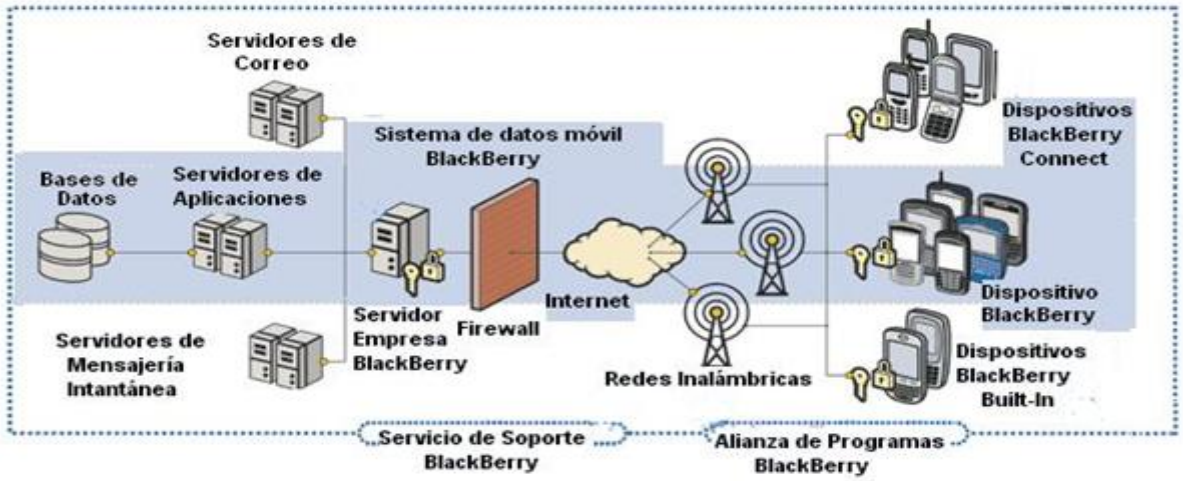

Figura 3 - Arquitectura de BlackBerry OS

En esta arquitectura se mantiene una estricta política de seguridad, que añade diferentes métodos de control en cada punto de conexión de los dispositivos. Se utilizan los mecanismos: AES (Advanced Encryption Standard) y TripleDES (Data Encryption Standard) para el cifrado de todo el tráfico de datos entre la aplicación y el servidor de la empresa BlackBerry, SSL/TLS (Secure Sockets Layer/ Transport Layer Security) para proteger la información que viaja del servidor de la empresa a los servidores de aplicaciones y PGP (Pretty Good Privacy) para asegurar los intercambios con la base de datos.

La arquitectura BlackBerry puede utilizarse asociada a los sistemas operativos AIX, BlackBerry OS, HP-UX, Solaris y Windows NT.

El IMS (Instant Message Service) es el framework de mensajería instantánea, el lenguaje de programación es Java, la máquina virtual es BlackBerry Java Virtual Machine y el perfil es MIDP (Mobile Information Device Profile).

Se dispone de las herramientas BlackBerry® JDE (Java® Development Environment),

BlackBerry Smartphone Simulator, API de Java ME y Java para BlackBerry para construir este tipo de aplicaciones, utilizando el lenguaje C++.

Los desarrolladores pueden crear con el BlackBerryIDE interfaces de usuario sofisticadas, para la entrada y búsqueda de datos compatibles con un subprocesamiento múltiple, internacionalización, comunicación de red y el almacenamiento local de datos. [18]

1.4.4 Arquitectura de iOS

La plataforma iOS creada por Apple, originalmente estaba concebida solo para los iphone, y en la actualidad se ha extendido hasta ser el soporte de dispositivos como el iPod, el iPad y los televisores Apple. Fue construida utilizando los conocimientos obtenidos de la creación de Mac OS X, por lo que muchas de las herramientas y tecnologías empleadas para desarrollar aplicaciones dentro de esta plataforma también tienen sus orígenes en dicho sistema operativo.

Es importante resaltar que la tecnología iOS solo puede ser utilizada en dispositivos Apple, ya que esta compañía mantiene un férreo control sobre el hardware, siendo la encargada de diseñar tanto el sistema iOS como el dispositivo donde se ejecuta

La interfaz de usuario está basada en el concepto de manipulación directa, usando tecnologías táctiles multi-gestos. Los elementos básicos de control de estas interfaces son los deslizadores, intercambiadores y botones.

La respuesta a las acciones del usuario es inmediata y con elegantes efectos visuales, ya que el sistema emplea acelerómetros internos para reaccionar ante movimientos como sacudidas o rotaciones del dispositivo (las imágenes en la pantalla rotan en tres dimensiones y cambian de posición vertical a horizontal automáticamente).

La versión actual del sistema operativo iOS 5.0.1 usa aproximadamente 770 megabytes de la capacidad total de almacenamiento del dispositivo, variando para cada modelo.

En la arquitectura iOS existen cuatro capas de abstracción: núcleo de iOS, núcleo de servicios, capa media y la Cocoa Touch, como se aprecia en la siguiente imagen:

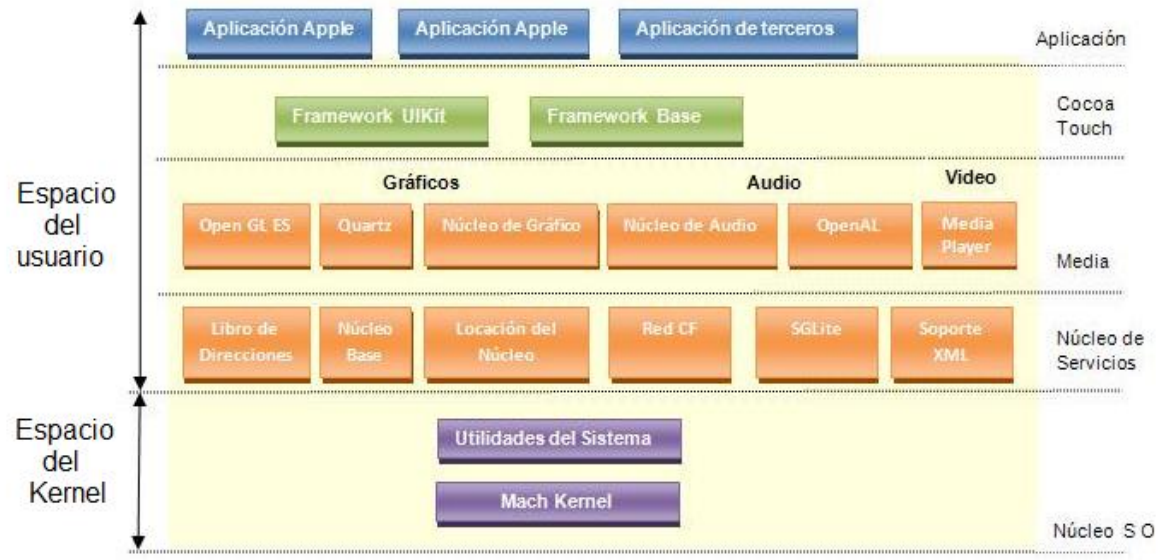

Figura 4 - Arquitectura de iOS

Para desarrollar una aplicación sobre esta plataforma se utilizan los lenguajes de programación C y C++, el IDE Xcode y las herramientas iOS SDK e iOS Simulator.

En esta arquitectura se contemplan frameworks para las funciones matemáticas (Accelerate), gestionar las cuentas de usuario del sistema (Accounts), para acceder directamente a la base de datos de contactos (AddressBook) y las bibliotecas multimedias (AssetsLibrary), para manejar archivos de audio (AudioToolbox), para acceder a redes inalámbricas (CFNetwork), para utilizar los servicios telefónicos (CoreTelephony), y otros que de manera general aprovechan todas las prestaciones de los dispositivos de Apple.

Utiliza protocolos HTTP, HTTPS, TCP/IP, UDP y WAP.

Con respecto a la seguridad en las últimas versiones se incluyen niveles adicionales para proteger los archivos, que garantizan el acceso a la información aunque el dispositivo esté bloqueado. En esta arquitectura las aplicaciones se diseñan para asegurar los datos desde que son creados y para preparar el acceso cuando el usuario bloquee o desbloquee el dispositivo. [19]

1.4.5 Arquitectura de Windows Phone 7

El 15 de febrero de 2010, durante el Mobile World Congress, Microsoft presentó al mundo su nuevo sistema operativo para teléfonos inteligentes: Windows Phone 7.

Esta tecnología estrena un nuevo concepto llamado HUB, un lugar donde centralizar acciones y agrupar aplicaciones por la actividad a la que se destinan, así, podemos

encontrar el HUB de Imágenes, Office o Xbox Live, cada uno de los cuales nos dará acceso a tareas específicas como música, documentos o juegos.

Windows Phone 7 está basado en Windows CE 6.0 R3, un sistema que soporta 32.768 hilos de proceso en su Kernel. La plataforma de aplicaciones reside en memoria de usuario mientras que el Kernel, los drivers, el sistema de archivos, el sistema de generación de gráficos y el sistema de actualizaciones residen en el espacio de Kernel, como se aprecia en la siguiente imagen:

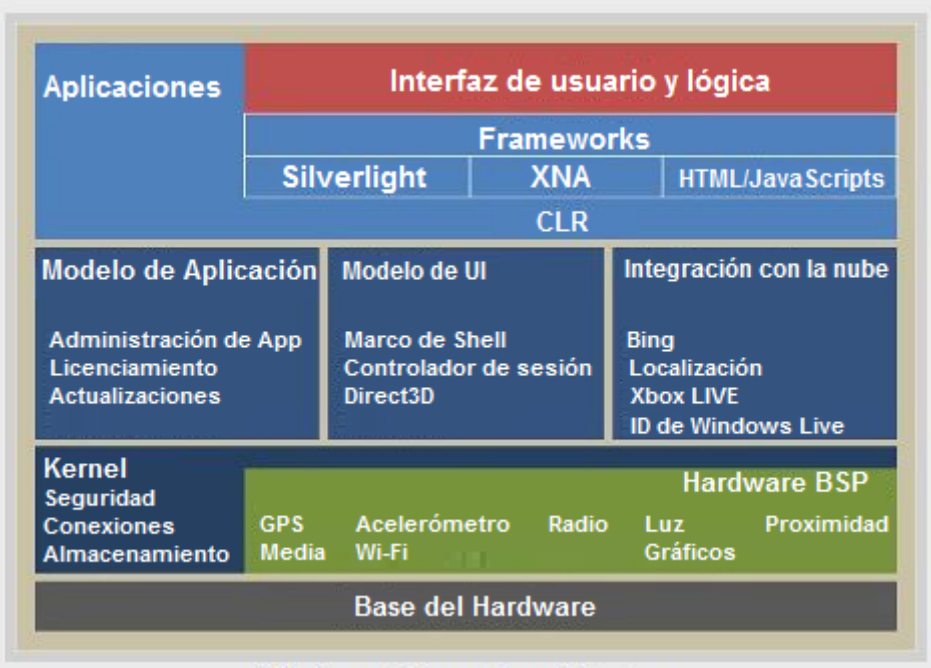

Figura 5 - Arquitectura de Windows Phone OS

El modelo de interfaz de usuario de Windows Phone 7 se compone de elementos, páginas y sesiones. Utiliza tecnología Silverlight Mobile, el motor de Internet Explorer 9, con su soporte para HTML5, multitarea en aplicaciones de terceros e integración con Xbox 360.Tiene integración con servicios como Exchange, Google Mail, Hotmail, Xbox Live, Skydrive, Facebook o Bing.

Presenta los frameworks para Windows Phone Silverlight y XNA. Estos frameworks se ejecutan sobre un sandbox de .NET que les facilita el acceso al hardware, sensores, almacenamiento, localización, etc.

Para desarrollar aplicaciones con esta tecnología Microsoft ha liberado el Windows Phone Cloud Services SDK, un kit de desarrollo que dotará de servicios exclusivos para Windows Phone 7, en el cual se incluyen las herramientas: Microsoft Visual Studio 2010 Express, Microsoft Expression Blend, Emulador de Windows Phone 7, Application Deployment y Windows Phone Developer Registration. Se emplea el lenguaje de programación C#. [20]

1.4.6 Comparación entre las arquitecturas de Android OS, BlackBerry OS, iOS y Windows Phone OS

A continuación se presenta una tabla comparativa entre las arquitecturas de los sistemas operativos Android OS, BlackBerry OS, iOS y Windows Phone OS:

Componente\Arquitectura	Android OS	BlackBerry OS	iOS	Windows Phone OS
Buscador	Google Browser	WAP	Safari	Internet Explorer
Compañía	Google	RIM	Apple	Microsoft Windows
Gráficos 3D	OpenGL	AirPlay, Unity 3	OpenGL	DirectX
IDE	Eclipse	BlackBerryIDE	Xcode	Microsoft Visual Studio
Interfaz de usuario	FW de Aplicaciones Java	GUI	Cocoa	Silverlight
Lenguaje de programación	Java	C++	C, C++	C#
Máquina virtual	Dalvik VM	BlackBerry Java VM	_	CLR
Tienda de aplicaciones	Android Market	BlackBerry App World	AppStore	App Marketplace

Tabla 1: Comparación entre las arquitecturas de los sistemas operativos Android OS, BlackBerry OS, iOS y Windows Phone OS

1.5 Fundamento de las aplicaciones compuestas

Una aplicación tradicional (también conocida como aplicación monolítica) contiene todo lo necesario para resolver determinados problemas del negocio, para la cual es creada y contiene los componentes que permiten en acceso y transformación de los datos contenidos en las bases de datos, los componentes para implementar tanto la lógica de negocio como la de proceso, y los componentes que implementan la interface que se le mostrará a los clientes.

En cambio ahora se puede desarrollar una aplicación compuesta, que no requiere implementar determinadas funcionalidades vinculadas a: el acceso a datos, la lógica de proceso, la lógica de negocio y los servicios de Seguridad.

Todas estas funcionales que pueden ser reutilizadas por otras aplicaciones son movidas hacia servicios contenidos dentro de una arquitectura de servicios que por sus características serán altamente reusables. Con el cambio de paradigma, de la orientación a objetos a la orientación a servicios, la forma de desarrollar las aplicaciones también cambia.

Las aplicaciones monolíticas tienen componentes para el acceso a datos, para la seguridad, y para muchas otras cosas que no pueden ser reusables. Un paso de avance son los frameworks, pero los elementos que se implementan dentro de ellos no son reutilizables en su mayoría a no ser que expresamente se diseñen e implementen para que lo sean. El alto acoplamiento entre clases y entre componentes en una misma aplicación impide que pueda ser reusados en otras aplicaciones.

En una aplicación compuesta los componentes para la presentación son los únicos propios de dicha aplicación que no presentan reutilización y los que consumirán las funcionalidades de los servicios. Y con eso se tiene acceso a servicios de acceso a datos, de negocio, de proceso, de seguridad, en fin, a todos los recursos ya implementados por otras aplicaciones y que forman parte del portafolio de servicios de la organización.[21]

1.6 Elementos de una aplicación compuesta

Una aplicación compuesta consta de módulos de acoplamiento flexible que se descubren y componen dinámicamente en tiempo de ejecución, estos módulos contienen componentes visuales y no visuales que representan al sistema. La siguiente imagen muestra la estructura general de este tipo de aplicaciones:

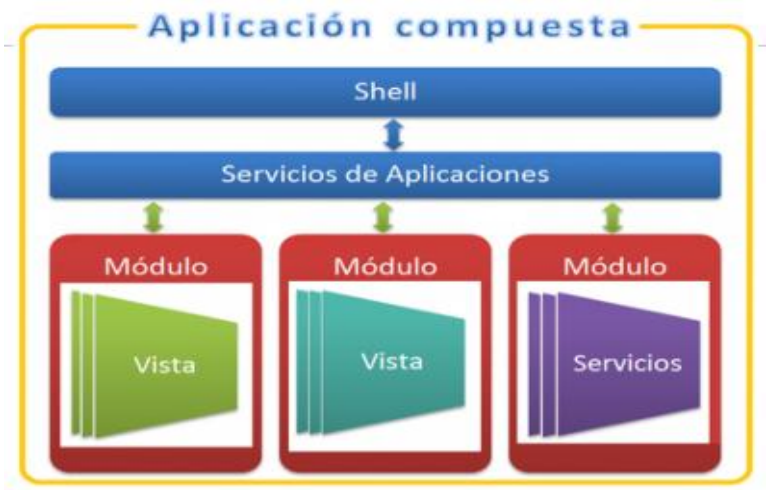

Figura 6 - Estructura de una aplicación compuesta

Los componentes visuales o vistas se componen en un Shell común que actúa como host de todo el contenido de la aplicación.

El Shell es la aplicación host dentro de la cual se cargarán los módulos, el Shell define la estructura de la aplicación. Por lo general, implementa servicios comunes de las aplicaciones y la infraestructura, pero la mayoría de la funcionalidad de la aplicación se lleva a cabo dentro de los módulos. El Shell también proporciona la ventana de nivel superior o

elemento visual que luego servirá como host de los distintos componentes de interfaz de usuario proporcionada por los módulos cargados.

Los módulos son paquetes de funcionalidad que pueden ser desarrollados y probados y desplegados de forma independiente. En muchos casos, los módulos son desarrollados y mantenidos por equipos separados. Se pueden utilizar para representar una funcionalidad específica relacionada con el negocio (por ejemplo, gestión de perfiles) y encapsular todos los puntos de vista, servicios y modelos de datos necesarios para implementar esa funcionalidad. Los módulos también se pueden utilizar para encapsular la infraestructura de aplicaciones o servicios comunes (por ejemplo, autenticación y servicios de gestión de excepciones) que pueden ser reutilizados en múltiples aplicaciones.

Las vistas son controles de interfaz de una función particular o área funcional de la aplicación. Las vistas se utilizan para proporcionar una separación entre la interfaz de usuario, la lógica de la aplicación y presentación de datos; además para definir el comportamiento de la interacción del usuario.

Los servicios son componentes que encapsulan la funcionalidad de la interfaz de usuario no relacionada, tales como autenticación, administración de excepciones, y acceso a datos. Los servicios pueden ser definidos por la aplicación o dentro de un módulo, y son a menudo registrados en el contenedor de inyección de dependencia para que puedan ser ubicados o construidos según sea necesario y utilizado por otros componentes que dependen de ellos. [22]

1.7 Aplicaciones compuestas en SOA

Service Oriented Architecture (SOA o Arquitectura Orientada a Servicios), es un estilo único de arquitectura y diseño de aplicaciones empresariales, que emplea servicios de negocio como bloque de construcción conceptual.

SOA ha alcanzado una enorme importancia para la industria del desarrollo de software y para las empresas que necesitan alinear sus departamentos de tecnología con lo que realmente hacen en el negocio, y esto se debe a la cantidad de escenarios de aplicación que posee y a la importancia de estos en la actualidad. [23]

El modo de construir aplicaciones en una arquitectura SOA cambia respecto al modelo tradicional. Dentro del nuevo paradigma las aplicaciones se construyen por medio de composición de servicios.

La pieza básica de una aplicación es un servicio. Los servicios se organizan a través de procesos de negocio que pueden estar orquestados en lenguaje BPEL (Business Process

Execution Language o Lenguaje de Ejecución de Procesos de Negocio). Estos procesos de negocio en sí mismos son servicios web.

La presentación de información al usuario se realiza mediante portlets de nueva generación, que soportan acceso remoto (WSRP: Web Service for Remote Portlet). Dichos portlets, proyectan su funcionalidad para ser consumida por un portal cliente que no necesita instalar absolutamente nada del portlet remoto y que además es capaz de aplicar su estilo gráfico local. En la siguiente imagen se observa la estructura de una aplicación compuesta en el entorno SOA [24]:

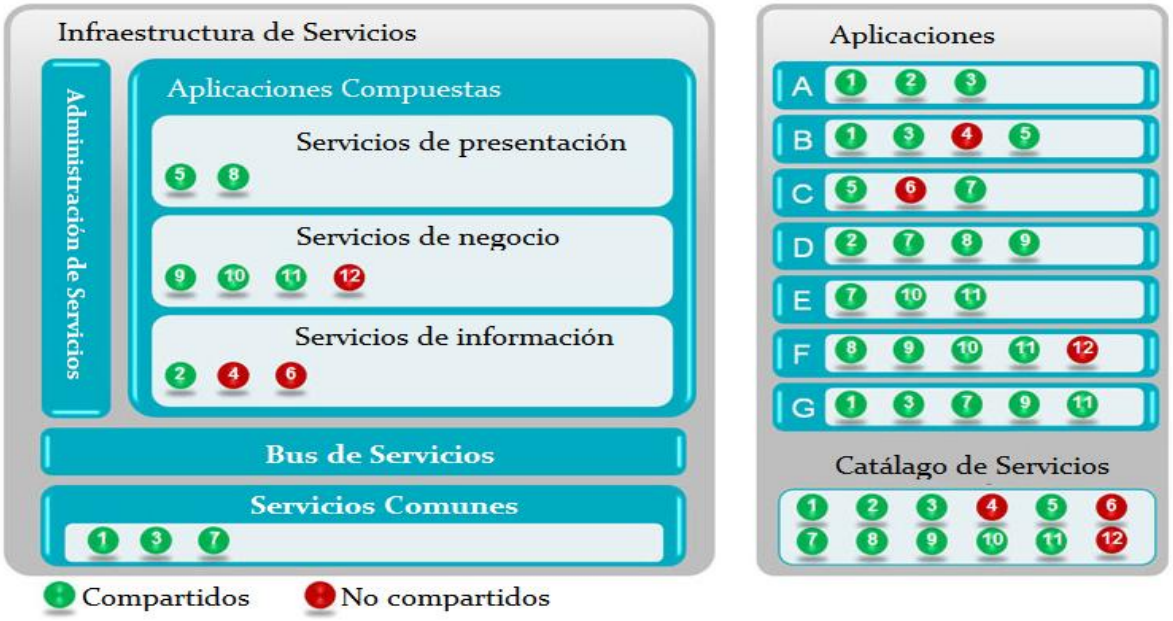

Figura 7 - Estructura de las aplicaciones compuestas en SOA

1.8 Fundamentos de Android

Android es una plataforma de desarrollo de aplicaciones móviles y además es un sistema operativo basado en el Kernel de Linux, orientado a dispositivos móviles tales como teléfonos inteligentes, ordenadores en tableta y PDAs.

Fue creado por Android Inc. y adquirido por Google Inc. que después se unió con Open Handset Alliance. Google ha publicado la mayoría del código fuente de Android bajo la licencia de Software Apache. El Anexo 1 muestra el surgimiento de la tecnología Android y las características de cada versión en la que ha evolucionado como sistema operativo.

1.9 Comparación entre Android y otros sistemas operativos para dispositivos móviles

Los creadores de la tecnología Android han desarrollado un sistema operativo que cualquier fabricante puede licenciar e incluir en su dispositivo, sin unos requerimientos mínimos de hardware ni límites en la personalización de la interface de usuario. Su principal ventaja es la cantidad, gracias a este modelo el mercado puede ser inundado con cientos de modelos distintos y el usuario tiene libertad para elegir entre terminales de gama baja, media o alta.

1.9.1 Android vs Blackberry OS

En el caso de Blackberry en la mayoría de los aspectos no supera a sus competidores. Existen poca cantidad de aplicaciones disponibles, no soporta múltiples cuentas de intercambio de mensajes, la captura y reproducción de videos realmente no es de buena calidad, como tampoco lo son la pantalla y su resolución para alta definición de imágenes, además que no permite videollamadas, además la plataforma está un poco desorganizada y suele tardar un tiempo en ofrecer las actualizaciones para los equipos. [25]

1.9.2 Android vs iOS

La gran limitante en este caso es el férreo control sobre el hardware, siendo Apple la encargada de diseñar tanto el sistema iOS como el dispositivo donde se ejecuta. La principal ventaja de este modelo es la adaptación total del sistema operativo al dispositivo, puesto que ambos han sido concebidos como partes de un todo, por lo tanto ofrecen la misma experiencia de uso para todos los consumidores. Otras desventajas importantes con respecto a Android son el coste de adquisición y la nula variedad de dispositivos en el mercado, solo existe uno y no hay opciones de elección.

1.9.3 Android vs Windows Phone 7 OS

Como fabricante del sistema, Microsoft requiere que todo teléfono que desee ejecutar Windows Phone 7 disponga de unas características mínimas, para asegurar la consistencia de todos los usuarios del sistema, a partir de estas características los fabricantes de software son libres de ampliarlas en algunos casos y están obligados a cumplirlas con exactitud en otros. En este modelo se unen las ventajas del modelo IPhone, todos los usuarios obtienen la misma experiencia de uso y los desarrolladores saben que su aplicación funcionará de forma idéntica en todos los dispositivos Windows Phone 7, y las del modelo Android, no hay limitante a un solo hardware, existen distintos dispositivos de varios fabricantes, todos con unas características mínimas comunes pero con suficientes diferencias como para sentir que el cliente escoja el terminal que más se ajuste a su gusto.

El problema para los programadores que deseen utilizar aplicaciones creadas por ellos mismos, es que la única forma de instalar una aplicación en Windows Phone 7 es mediante la tienda oficial de Microsoft: Marketplace; en la cual debemos registrarnos como desarrolladores para poder vender nuestras aplicaciones. Es decir si Microsoft no compra la aplicación de determinado desarrollador, este nunca podrá utilizarla, ni siquiera en su propio dispositivo.[25]

A continuación se observa una tabla comparativa entre los sistemas operativos antes mencionados, donde la presencia del icono representativo de cada uno, indica su superioridad sobre los otros en determinado aspecto:

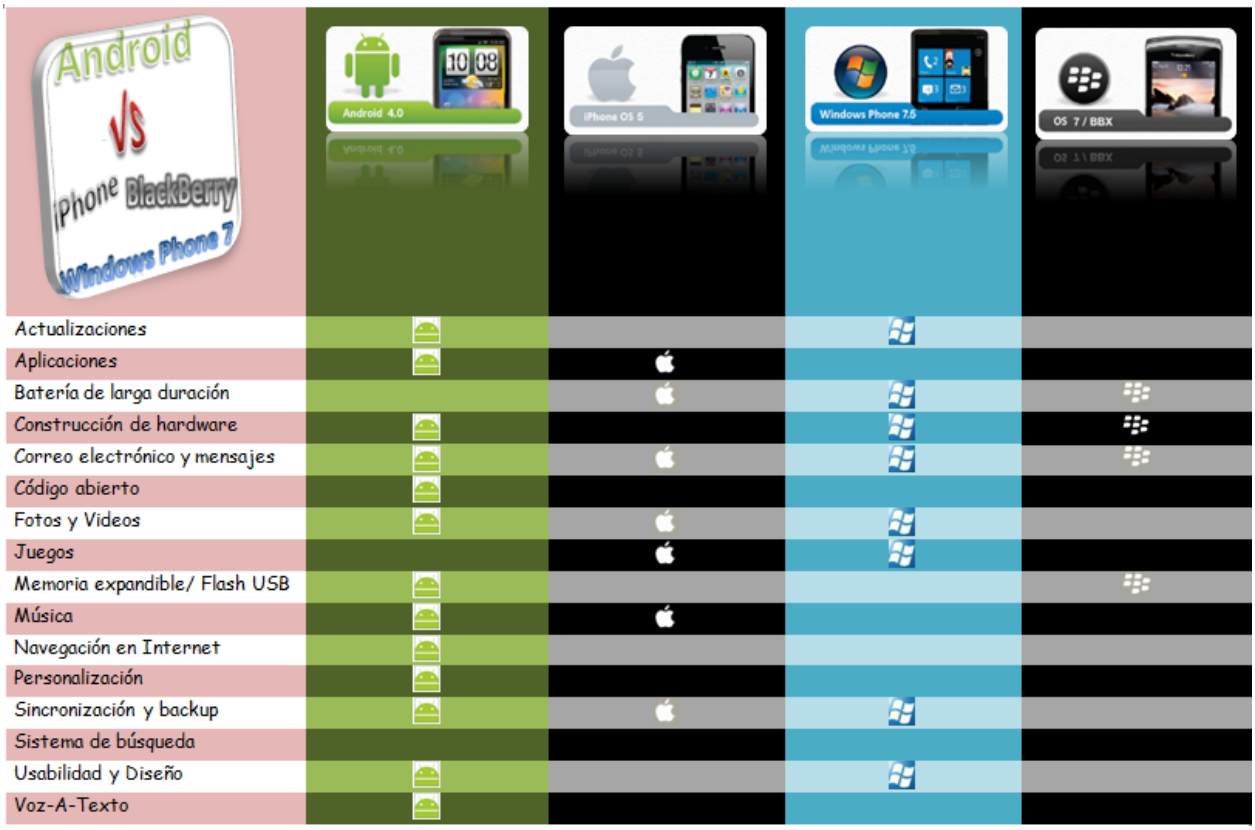

Figura 8 - Tabla comparativa entre Android OS, iOS, Windows Phone 7 y BlackBerry OS.

1.10 Conclusiones parciales

En este capítulo se demostró la factibilidad de Android como sistema operativo con excelentes posibilidades para permitir el desarrollo de aplicaciones en la Universidad de las Ciencias Informáticas. Se analizaron las características de las aplicaciones compuestas y su alta compatibilidad con los principios de la estrategia SOA, que a su vez establece las directrices investigativas y productivas en el CDAE. Además se compararon diferentes arquitecturas de referencia, a fin de seleccionar algunos elementos que puedan ser de utilidad para el diseño de la propuesta arquitectónica en la que se enmarca este trabajo.

Capítulo 2 Propuesta arquitectónica de aplicaciones compuestas para Android.

2.1 Introducción

En este capítulo se presenta la solución al problema planteado, una propuesta arquitectónica factible para el desarrollo de aplicaciones compuestas para dispositivos móviles que utilicen el sistema operativo Android. Dicha arquitectura estará orientada a las características específicas del entorno móvil, adaptada a las condiciones de la Universidad de las Ciencias Informáticas y aprovechará las bondades de Android. Se plantea el empleo de las redes de conexión inalámbrica disponibles en la universidad, pues la red GSM del proveedor nacional ETECSA no ofrece los servicios de acceso y navegación en Internet. Esta arquitectura incluye una capa para el empleo del hardware adicional en los dispositivos tale como acelerómetros, cámara de fotos y videos, giroscopios, localizadores de posicionamiento global, magnetómetros, pantallas táctiles, sensores de proximidad y termómetros.

2.2 Formato de mensajería

La Empresa de Telecomunicaciones de Cuba S.A (ETECSA) comercializa la telefonía móvil en sus modalidades prepago y pospago a personas naturales y jurídicas, a través de su servicio Cubacel, operando la norma GSM (900 MHz) con cobertura nacional. La compañía permite el envío y recepción de mensajes en dos formatos: [26]

- SMS: Servicio que permite a los usuarios prepago y pospago el envío y la recepción de mensajes de texto corto nacionales e internacionales. La tarificación de los mensajes se realiza por cada SMS enviado, independientemente de que estos sean recibidos o no por el destinatario. Si el mensaje sobrepasa los 160 caracteres, este se divide en más de uno, y se cobran todos los mensajes que resulten al final, cada mensaje a lo sumo puede tener 160 caracteres.
- MMS: Es un servicio que ofrece la posibilidad de enviar y recibir mensajes con contenidos de texto, imágenes, audio y video. Está disponible para todos los usuarios del servicio de telefonía móvil. Con un tamaño máximo de 260KB por envío, la facturación es por evento, con independencia del volumen de datos transmitidos. Los mensajes multimedia se pueden enviar solo a los usuarios de Cubacel, el envío internacional (a móviles de otros operadores) no está disponible.

2.3 Frameworks de aplicaciones

Los desarrolladores de aplicaciones para Android, disponen de acceso total al código fuente usado en las aplicaciones base. Esto evita que se generen cientos de componentes pertenecientes a diferentes programas, los cuales respondan a la misma acción, brindando a cualquier usuario la posibilidad de modificar o reemplazar elementos de los programas sin tener que desarrollar sus aplicaciones desde el principio.[27]

Construir aplicaciones utilizando frameworks otorga al programador ventajas tales como el ahorro de código, la agilización y la simplificación del proceso de desarrollo.[28] Los frameworks pueden incluir soporte para sistemas, librerías y lenguajes de programación, algunos ejemplos para el desarrollo de aplicaciones compuestas sobre la plataforma Android son:

- ✓ Titanium: es de código abierto, permite crear aplicaciones con tecnología web para iPad, iPhone y Android. Contiene Titanium+Plus Modules, que son extensiones para poder invocar código nativo Objective-C o Android mientras se ejecuta línea a línea el JavaScript del proyecto. Además permite utilizar HTML5 (HyperText Markup Language o Lenguaje de Marcado de Hipertexto) y CSS (Cascading Style Sheets u Hojas de estilo en cascada).[29]
- ✓ PhoneGap: es una implementación a código abierto de estándares libres, se utiliza para desarrollar aplicaciones para iPhone, BlackBerry, Android y Nokia S60 mediante el uso de JavaScript y HTML5. Al emplear tecnologías basadas en estándares web para unificar las aplicaciones web y los dispositivos, PhoneGap podrá trabajar con los navegadores independientemente de cuánto evolucionen.[30]
- ✓ Rhodes: está basado en el lenguaje de programación Ruby, es de código abierto, genera de forma agilizada aplicaciones nativas para iPhone, iPad, BlackBerry, Android, Windows Mobile y Symbian, mediante HTML, CSS y Javascript. Soporta la estructura Modelo-Vista-Controlador en la organización de las clases y la lógica del negocio.[31]
- ✓ Look!: pretende aunar en un sólo framework funcionalidades básicas requeridas en el desarrollo de aplicaciones de realidad aumentada. Contiene una galería de imágenes en 3D, un mundo virtual, un juego interactivo en tres dimensiones y una aplicación para la creación de redes sociales con soporte para geo localización. Adicionalmente, se han escrito tutoriales que asistan al uso de este framework, y se ha documentado su funcionamiento por si otros equipos quisieran continuar con este desarrollo.[32]

2.4 Herramientas libres

Los desarrolladores de aplicaciones para dispositivos móviles pueden utilizar diversas herramientas de software libre, ya sean IDEs, SDKs, simuladores o emuladores. Algunos ejemplos son:

- ✓ Eclipse IDE: Entorno de desarrollo multiplataforma. Permite utilizar varios lenguajes de programación como C++, Python, Java. Consume algunos recursos del sistema en tiempo de ejecución. [33]
- ✓ NetBeans IDE: Entorno de desarrollo multiplataforma. Permite utilizar varios lenguajes de programación como C++, Ruby, Php, Python, Javascript; así como desarrollar aplicaciones móviles. [33]
- ✓ Android SDK: provee las herramientas y APIs necesarias para desarrollar aplicaciones sobre la plataforma de Android utilizando el lenguaje de programación Java. [17]
- ✓ Android Development Tools (ADT): es un plugin de desarrollo para el IDE de Eclipse. Extiende las capacidades de Eclipse para permitirte configurar rápidamente nuevos proyectos de Android, crear una interfaz de usuario de la aplicación, añadir componentes en función de la API de Android Framework, depurar tus aplicaciones mediante las herramientas de Android SDK e incluso exportar APKs con (o sin) firma con el fin de distribuir la aplicación iglú . [34]

2.5 Patrón de arquitectura

Para el desarrollo de aplicaciones compuestas para dispositivos móviles sobre la plataforma Android se propone el uso del patrón arquitectónico N-Capas, muy recomendable ya que los entornos de ejecución de dichas aplicaciones estarán distribuidos en diferentes elementos o nodos tanto lógicos como físicos. Aquí aparecen dos conceptos importantes: las capas (Layers) que se encargan de la distribución lógica de los componentes, y los niveles (Tiers) que se refieren a la colocación física de los recursos. La característica principal de este patrón es que cada capa oculta las capas inferiores de las siguientes superiores a esta.

Algunas de las ventajas de utilizar este patrón son: [35]

- ✓ Mejoras en las posibilidades de mantenimiento, debido a que cada capa es independiente de la otra los cambios o actualizaciones pueden ser realizados sin afectar la aplicación como un todo.
- ✓ Escalabilidad, ya que las capas están basadas en diferentes máquinas, el escalamiento de la aplicación hacia afuera es razonablemente sencillo.

- ✓ Flexibilidad, como cada capa puede ser manejada y escalada de forma independiente, la flexibilidad se incrementa.
- ✓ Disponibilidad, las aplicaciones pueden aprovechar la arquitectura modular de los sistemas habilitados usando componentes que escalan fácilmente lo que incrementa la disponibilidad.

La descomposición en capas que se propone es:

1. Capa de presentación: es la única que interactúa directamente con el usuario presentándole el sistema, permitiendo el intercambio de información entre ambos. Contiene una interfaz gráfica que posibilita capturar los datos insertados por los clientes, además de mostrarles los resultados de sus peticiones y los estados de la aplicación. Esta capa solo interactúa con la capa de negocio, que es su inferior inmediata.
2. Capa de negocio: también conocida como lógica de negocio, es la que recibe las peticiones de la capa de presentación y le envía las respuestas tras el proceso. Aquí se realiza la mayor parte del procesamiento de la información del dominio de la aplicación, se ejecutan cálculos sobre los datos de entrada o almacenados, se validan los contenidos provenientes de la capa superior y se ejecutan los algoritmos específicos del programa. Contiene un componente propio del sistema operativo Android cuya función principal es manejar el consumo de servicios web. Esta capa interactúa con la capa de presentación, para recibir sus solicitudes y presentarle los resultados; y con la capa de datos, para solicitar al gestor de base de datos almacenar u obtener datos de él.
3. Capa de acceso a datos: es la encargada de intercambiar con otros sistemas, solo mediante ella se puede acceder a los recursos obtenidos de terceras aplicaciones. Contiene uno o más gestores de base de datos para lenguaje SQL Esta capa interactúa con la capa de negocio, recibe sus solicitudes de almacenamiento o recuperación de información, y envía la respuesta a las peticiones.

Con el objetivo de exponer claramente el funcionamiento de la propuesta arquitectónica para el desarrollo de aplicaciones compuestas para dispositivos móviles con el sistema operativo Android, se presentan los siguientes diagramas que muestran la estructura de la arquitectura, los elementos que la componen y la forma que interactúan entre ellos, se han modelado en una escala que va de lo general a lo específico:

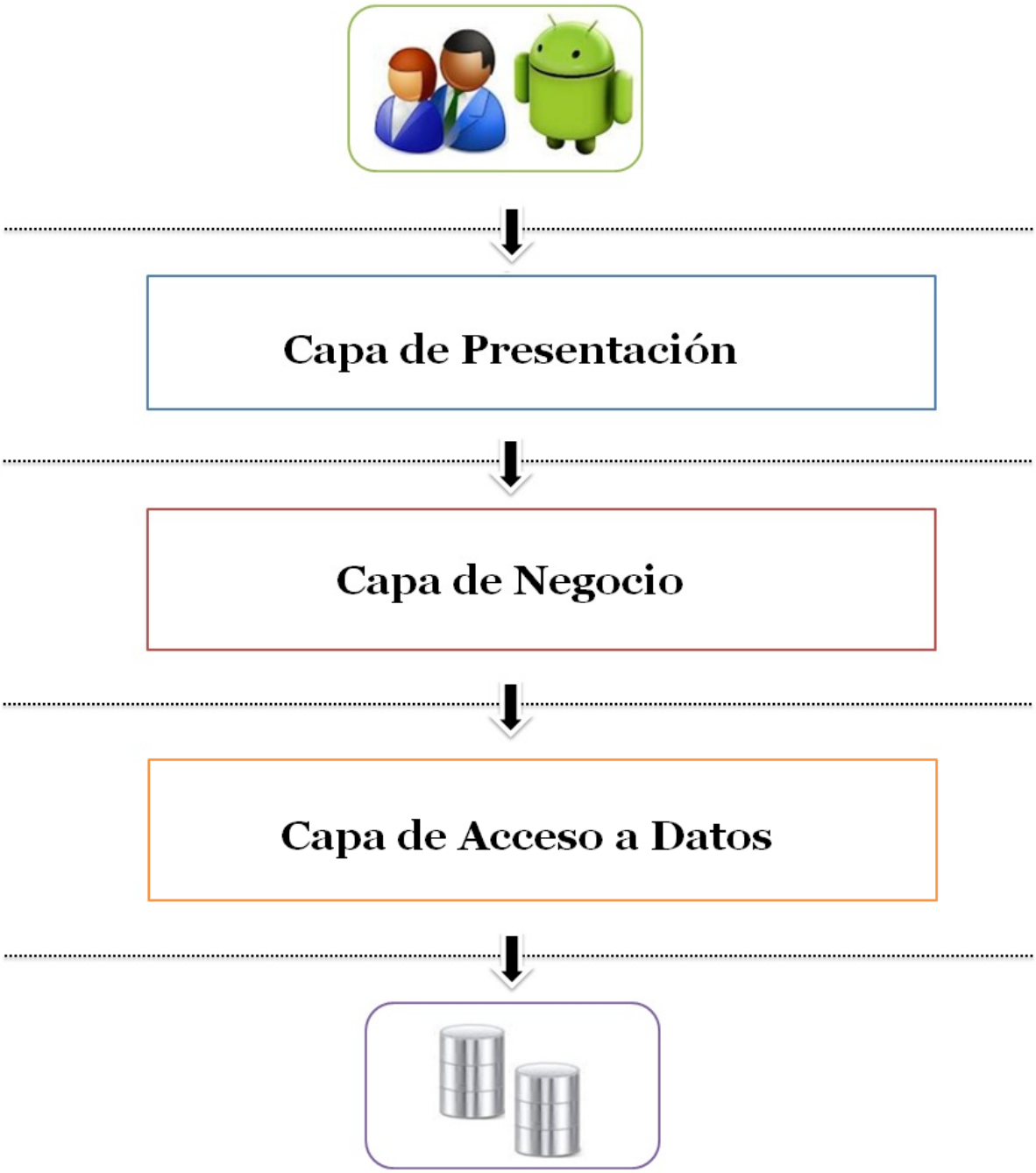

Figura 9 Diagrama general de la propuesta arquitectónica

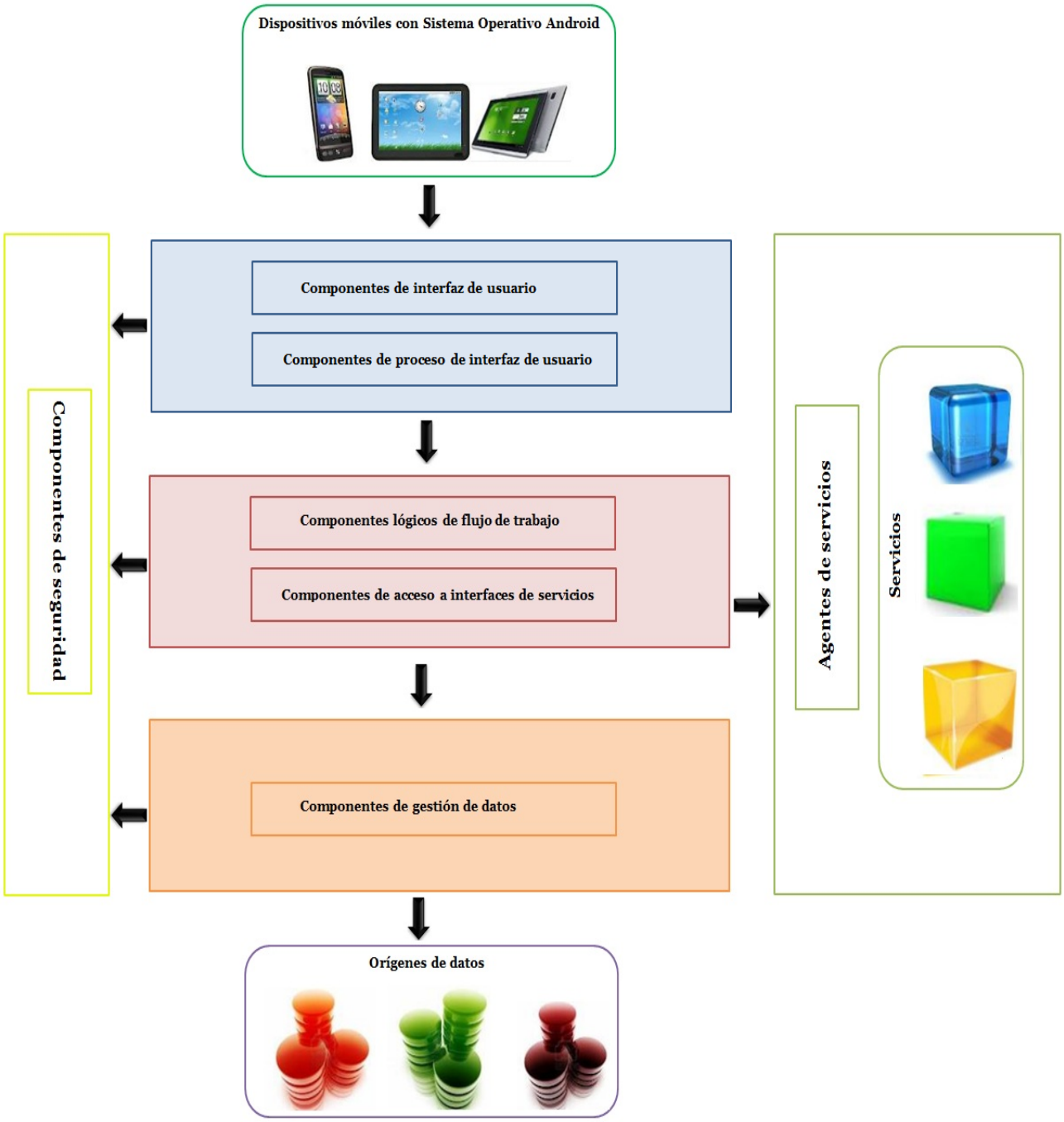

Figura 10 - Diagrama orientado a componentes de la propuesta arquitectónica

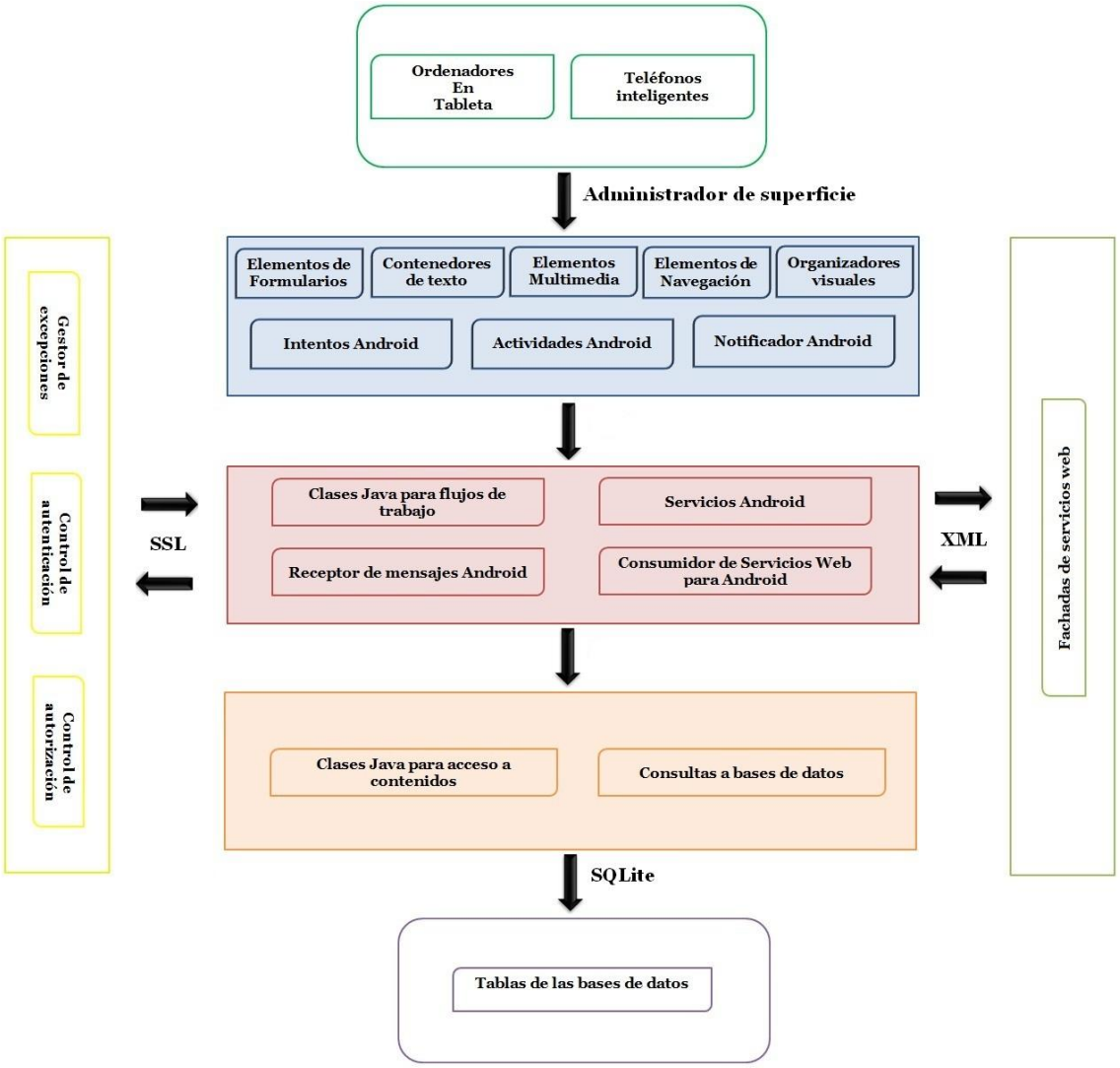

Figura 11 - Diagrama orientado a dominio de la propuesta arquitectónica

A continuación se describen los elementos de cada capa:

1) Capa de presentación
    a) Componentes de interfaz de usuario
        i) Elementos de formularios: tienen diseños nativos del sistema operativo Android, incluyen botones, etiquetas, selectores, barras de progreso y de puntuación, etc.
        ii) Contenedores de texto: posibilitan capturar cadenas de caracteres insertadas por el usuario, ya sea en forma de texto plano o como contraseñas ocultas.
        iii) Elementos multimedia: permiten reproducir audio y visualizar vídeos e imágenes en el dispositivo móvil.

iv) Elementos de navegación: posibilitan embeber sitios web como parte de las aplicaciones.

v) Organizadores visuales: son una especie de cajas contenedoras que distribuyen en la interfaz gráfica los restantes componentes de presentación al usuario.

b) Componentes de proceso de interfaz de usuario

i) Actividades Android: representan la capa de presentación de toda aplicación Android, por ejemplo, una pantalla que el usuario ve. Una aplicación para Android puede tener varias actividades y se puede cambiar entre ellas en tiempo de ejecución de la aplicación.

ii) Intentos Android: una aplicación Android puede tener varias actividades, para cambiar de una a otra, ya sea que se envíe o no información entre ellas, la plataforma solo permite hacer la transición a través de la Interfaz de Intención.

iii) Notificador Android: engloba los servicios para notificar al usuario cuando algo requiera su atención mostrando alertas en la barra de estado. [36]

2) Capa de negocio

a) Componentes lógicos de flujos de trabajo

i) Clases Java para flujos de trabajo: se encargan de ejecutar las tareas propias de la aplicación, en ellas se realizan cálculos, se validan y convierten al formato correcto los datos provenientes de la capa de presentación.

ii) Servicios Android: realizan tareas propias del sistema operativo en un segundo plano, sin ofrecer una interfaz de usuario.

iii) Receptor de mensajes Android: recibe los mensajes del sistema y las solicitudes implícitas, se puede utilizar para responder a condiciones cambiantes en el sistema.

b) Componentes de acceso a interfaces de servicios

i) Consumidor de servicios web para Android: encapsula los datos de la petición para consumir determinado servicio (nombre, dirección de publicación, métodos), y obtiene la respuesta de la fachada para luego ser procesada por la aplicación.

3) Capa de acceso a datos

a) Componentes de gestión de datos

i) Clases Java para acceso a contenidos: encapsulan los datos que se compartirán entre aplicaciones para tener control sobre cómo se accede a la información. [36]

ii) Consultas a bases de datos: permiten insertar, eliminar y modificar datos contenidos en las tablas de las bases de datos a las que accede la aplicación.

Otros elementos serían:

1) Seguridad
   a) Gestor de excepciones: maneja los problemas ocurridos en tiempo de ejecución, no permite al usuario interactuar directamente con los errores del sistema.
   b) Control de autenticación: evita el acceso a la aplicación por parte de personal no autorizado, se aplica mediante la inserción de la combinación usuario y contraseña para entrar al sistema.
   c) Control de autorización: restringe el acceso a determinados escenarios e información, según el rol del usuario que se ha autenticado en el sistema.
2) Servicios
   a) Fachadas de servicios web: es lo que publican los servicios para que las aplicaciones puedan consumir datos de ellos.

Las tecnologías utilizadas son:

1) Administrador de superficie: organiza lo que se mostrará en pantalla. En esencia crea las superficies en la pantalla que posteriormente pasarán a ser ocupadas por las actividades. [36]
2) SQLite: SQLite es una librería de software en lenguaje C, que implementa un sistema autónomo, sin servidor, sin necesidad de configuración y con el motor transaccional de base de datos SQL. El código fuente de SQLite es de dominio público. [37]
3) SSL: es la capa de conexión segura, establece un enlace protegido entre el sitio web y el navegador de la persona que lo visita, de forma que toda la información transmitida quede cifrada. Este sistema de protección se usa para transmitir mensajes de correo electrónico, archivos y todo tipo de información de forma segura. [38]
4) XML: es un Lenguaje de Etiquetado Extensible muy simple, pero estricto que juega un papel fundamental en el intercambio de una gran variedad de datos. Es similar a HTML pero su función principal es describir datos y no mostrarlos como es el caso de HTML. XML es un formato que permite la lectura de datos a través de diferentes aplicaciones, sirve para estructurar, almacenar e intercambiar información. [39]

2.6 Patrones de diseño

Un patrón es una unidad de información que expresa la relación entre tres componentes esenciales: un problema a resolver, una solución y un determinado contexto donde se repite una situación que vincula a los dos primeros componentes. Un conjunto importante de patrones de diseño son los publicados en el libro "Design Patterns", escrito por los que comúnmente se conoce como Gang of Four (GoF, en español "pandilla de los cuatro") formada por Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides. Ellos recopilaron y documentaron 23 patrones de diseño aplicados usualmente por expertos diseñadores de software orientado a objetos.

Siguiendo el libro de GOF los patrones se clasifican según el propósito para el que han sido definidos, del total se seleccionaron aquellos que se adecuan al entorno de desarrollo de aplicaciones compuestas con Android: [40]

- Creacionales: solucionan problemas de creación de instancias. Ayudan a encapsular y abstraer dicha creación.

  Constructor (Builder): separa la construcción de un objeto complejo de su representación, de forma que el mismo proceso de construcción pueda crear diferentes representaciones. En el caso de Android permite generar objetos globales que puedan ser utilizados en cualquier momento e indistintamente por todas las vistas y las clases del negocio del sistema.

- Estructurales: solucionan problemas de composición de clases y objetos.

  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 expandible, que requieren de un adaptador para crear los grupos padres y los hijos, que han de ser mostrados al usuario.

  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 aquel 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.

- De Comportamiento: soluciones respecto a la interacción y responsabilidades entre clases y objetos, así como los algoritmos que encapsulan.

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.

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

Observador (Observer): define una dependencia de uno-a-muchos entre objetos, de forma que cuando un objeto cambia de estado se notifica y actualizan automáticamente todos los objetos. Para este caso, se evidencian las bondades de este patrón cuando al modificar las instancias de los objetos globales, todas las variables dependientes cambian de forma automática.

2.7 Protocolos de comunicación con servidores

Los programas en el entorno móvil constantemente intercambian datos con sus servidores de correo, de archivos y de aplicaciones. Para definir la forma en que se realizarán las peticiones al servidor y en que se enviarán las respuestas a las solicitudes de los clientes, se establecen protocolos de comunicación. Entre los protocolos recomendados para los dispositivos móviles con sistema operativo Android se encuentran:

- ✓ IMAP (Internet Message Access Protocol o Protocolo de Acceso a Mensajes de Internet): ofrece una comunicación bidireccional entre las cuentas web de Gmail y el cliente de correo electrónico, esto significa que las acciones ejecutadas en los dispositivos móviles se reflejarán automáticamente en Gmail; esta sincronización evita la pérdida de los nuevos mensajes y la descarga repetida del mismo mensaje.[41]
- ✓ FTP: para garantizar el intercambio de archivos en el entorno móvil la implementación de este protocolo parte del uso de un cliente FTP en los terminales con Android, pudiera utilizarse AndFTP 2.0 o Bluetooth File Transfer 5.0 que establecen una conexión FTP vía bluetooth permitiendo explorar la estructura de directorios de otro terminal, subir o descargar archivos, editar archivos en memoria e incluso borrarlos.[42]

- ✓ HTTPS: comunica de forma segura los navegadores web en los dispositivos móviles con los servidores de Internet, protegiendo los datos en los envíos de solicitudes y respuestas. Está disponible a partir de la versión 2.3.4 de Android.
- ✓ 2G (2 Generación): usan una codificación digital e incluyen GSM, D-AMPS (TDMA) y CDMA. Estos protocolos soportan comunicaciones de voz de alta velocidad de bits y datos limitados. Ofrecen servicios auxiliares como datos, facsímil y SMS. La mayoría de los protocolos 2G ofrecen distintos niveles de cifrado y están dentro de las bandas de 880-915 MHz, 925-960 MHz, 1710-1785 MHz o 1805-1880 MHz. [43]
- ✓ 3G (3 Generación): soportan velocidades de datos mucho más altas, medidas en Mbps, planeadas para aplicaciones distintas a las de voz. 3G soporta aplicaciones que requieren de gran ancho de banda como video en movimiento, video conferencia y acceso completo a Internet. Las redes 3G pueden transmitir datos inalámbricos a 144 kilobits por segundo a velocidades de usuarios móviles. [43]

2.8 Premisas para el uso de la propuesta arquitectónica

En este acápite se plantean algunas proposiciones para aumentar la calidad de las aplicaciones compuestas desarrolladas sobre la plataforma Android y mantener buenas prácticas durante el uso de la arquitectura presentada en este trabajo.

2.8.1 Portabilidad

La portabilidad es la propiedad de los programas informáticos que permite su uso en diversos sistemas operativos compatibles.[44] En el caso de Android, esta proporciona una capa de abstracción completa, haciendo entonces que todas las aplicaciones sea portables, independientemente del hardware subyacente.[45] Para que una aplicación Android sea interoperable dentro de la plataforma, el desarrollador debe especificar mediante la SDK utilizada cuál será la versión mínima sobre la que funcionará dicha aplicación. Un software creado a partir de cualquiera de las generaciones Android será compatible y completamente funcional en todas las generaciones siguientes.

2.8.2 Principios de diseño de servicios en SOA

En la plataforma de computación orientada a servicios existen un conjunto definido de elementos que se aplican colectivamente para alcanzar los objetivos de las empresas. Dentro de dichos elementos están los principios de diseños de servicios, que son buenas prácticas para proponer una forma de lograr algo basadas en las experiencias pasadas o en la aceptación de toda la industria. Cuando se trata de soluciones de construcción, un

principio de diseño representa una muy recomendable guía para la configuración de la lógica de la solución de una determinada manera y con ciertos objetivos en mente.

A continuación se proponen algunos principios de diseño que contribuirán al aumento de la interoperabilidad intrínseca de las aplicaciones para dispositivos móviles con Android, así como al incremento de las opciones de proveedores de contenidos para dichas aplicaciones y la mejoría respecto a la agilidad organizacional: [46]

- ✓ Contratos de Servicios (estandarización y diseño): son la expresión consistente de las capacidades y el propósito general del servicio; establece las condiciones de contratación, proporcionando las limitaciones y requisitos técnicos, así como cualquier información semántica el propietario del servicio desea hacer pública.
- ✓ Acoplamiento de Servicios (intra-servicio y dependencias del consumidor): enfatiza en la comprensión de cómo medir y asignar niveles apropiados de acoplamiento de entre las partes de una solución orientada a servicios.
- ✓ Abstracción de Servicios (ocultación de información y tipos de meta-abstracción): evita la proliferación innecesaria de la información del servicio, los datos correspondientes a la lógica del servicio y su implementación no se publican pues protegen la integridad de la unión formada entre él y sus futuros consumidores.
- ✓ Reutilización de Servicios (diseño comercial y agnóstico): aboga por la reutilización repetida, se esfuerza por obtener el máximo valor posible de cada pieza de software.
- ✓ Autonomía de Servicios (procesamiento y control de fronteras): representa la capacidad de autogobernarse. Algo que es autónomo tiene la libertad y el control para tomar sus propias decisiones sin necesidad de aprobación externa. Por lo tanto, el nivel en que un servicio es autónomo representa la medida en que es capaz de actuar independientemente.
- ✓ Descubrimiento de Servicios (Interpretación y comunicación): ayuda a determinar si la automatización de los requisitos que se necesita cumplir ya existe dentro de un inventario de servicios.
- ✓ Componibilidad de Servicios (diseño de miembros de composición y composiciones complejas): introduce nuevas consideraciones de diseño que aseguran que los servicios son capaces de participar en múltiples composiciones para resolver varios problemas mayores; el ensamblaje de las capacidades de diferentes fuentes para resolver un problema más grande es la base de la computación distribuida.

2.8.3 Rendimiento

Para utilizar dispositivos móviles con Android es necesario tener en cuenta la velocidad de procesamiento limitada, el almacenamiento reducido, el ancho de banda restringido y las conexiones con alto nivel de latencia, además de la corta vida de las baterías. [47]

Las siguientes directrices son para mejorar el rendimiento de las aplicaciones para Android:

- Diseñar opciones configurables para permitir que se aprovechen al máximo las capacidades de los dispositivos. Permitir que los usuarios desactiven funciones que no necesiten, para así ahorrar energía.
- Tener en cuenta los recursos de memoria limitados y optimizar la aplicación para que use la cantidad mínima de memoria posible.
- Equilibrar el rendimiento con el consumo mínimo de energía presente al usar la CPU del dispositivo, la comunicación inalámbrica, la pantalla u otros recursos que afecten la duración de las baterías.

2.8.4 Seguridad

El diseño de una estrategia efectiva para la autenticación y autorización a los usuarios del sistema es importante para la seguridad y la fiabilidad de su aplicación. Una autenticación débil puede dejar a la aplicación a usos no autorizados. Los dispositivos móviles son generalmente diseñados para tener un solo usuario y, normalmente, carecen de perfil de usuario básico y el seguimiento de la seguridad más allá de una simple contraseña.

La exhibición de los dispositivos móviles a través de comunicaciones inalámbricas puede provocar escenarios inesperados. Deben ser consideradas las siguientes pautas para el diseño de la autenticación y autorización en las aplicaciones para Android: [48]

- ✓ Diseñe la autenticación para over-the-air (sobre-el-aire), la sincronización de contenidos y escenarios locales en la tarjeta de almacenamiento.
- ✓ Recuerde la posibilidad de que los diferentes dispositivos tengan variaciones en sus modelos de programación de seguridad, esto puede afectar la autorización para acceder a los recursos.
- ✓ Utilice las listas de control de acceso (ACL) del sistema operativo para mejorar el nivel de seguridad de archivos.
- ✓ Asegúrese de que usted requiera de autenticación para el acceso de los dispositivos por vía Bluetooth.

2.8.5 Usabilidad

Para mejorar respecto a la usabilidad de las aplicaciones compuestas desarrolladas para dispositivos móviles con Android se recomienda:

- ✓ Evitar el consumo excesivo de los recursos de los terminales; el uso de grandes cantidades de memoria volátil o demasiados requerimientos sobre procesador de información puede incidir de forma negativa en el rendimiento del terminal (disminuyendo la velocidad de respuesta del teclado táctil, restando calidad a la visualización de imágenes, etc.).
- ✓ Limitar las tareas en segundo plano; se aprovecha mejor la capacidad multitarea del sistema operativo Android si se elimina el proceso de las aplicaciones que el usuario ya no esté utilizando.
- ✓ Incluir en el menú de contexto de la aplicación la opción de eliminar su proceso propio.
- ✓ Almacenar los datos generados por la aplicación en la tarjeta de memoria externa, no en la del terminal móvil. En caso de que la capacidad de almacenamiento adicional esté agotada, brindar la opción de liberar espacio.
- ✓ Garantizar un entorno visual sencillo y agradable; fácil de comprender y usar; con un diseño ergonómico.
- ✓ Actualizar constantemente, brindando a los clientes correcciones de posibles errores y mejorías notables en cada nueva versión.
- ✓ Mantener la aplicación disponible a cada momento.
- ✓ Documentar la aplicación, puede crear un manual de usuario aparte o una sección de ayuda contenida en el programa.

2.9 Conclusiones parciales

En este capítulo se completó la propuesta arquitectónica para el desarrollo de aplicaciones compuestas para dispositivos móviles con sistema operativo Android. La misma consta de varios aspectos que facilitarán el proceso de construcción de software a los equipos de trabajo que utilicen esta plataforma, pues incluye herramientas y frameworks altamente compatibles con dicha tecnología; patrones diseño útiles para asignar responsabilidades a los objetos y resolver problemas recurrentes en nuevos contextos; además define directrices para garantizar el rendimiento, la usabilidad y la seguridad en el entorno móvil.

Capítulo 3 Validación de la propuesta arquitectónica

3.1 Introducción

En esta etapa se realiza la validación de la propuesta arquitectónica para el desarrollo de aplicaciones compuestas para dispositivos móviles que utilicen el sistema operativo Android. Se emplean dos estrategias, la primera es el uso del método de Iadov para valorar la satisfacción de un grupo de expertos y la segunda es la implementación de un caso de estudio práctico sobre un escenario real en la Universidad de las Ciencias Informáticas.

3.2 Validación por expertos utilizando la Técnica V.A Iadov.

La “Técnica V. A. Iadov” fue creada para establecer el nivel de satisfacción por la profesión de carreras pedagógicas, en este caso se ha reconducido la técnica para poder aplicarla a las actividades en el medio de desarrollo de aplicaciones compuestas para Android. [49]

Se seleccionó esta técnica para la validación de la propuesta arquitectónica del presente trabajo de diploma pues constituye una vía indirecta para el estudio de la satisfacción de un grupo de personas a la hora de desempeñar ciertas actividades. El índice de satisfacción grupal obtenido luego de procesar todas las respuestas de los entrevistados, es una de las bondades de esta práctica. Se basa en el análisis de un cuestionario que tiene una estructura interna determinada que desconoce el entrevistado. Se ha modificado el cuestionario para que pueda ser aplicado en el campo de las Arquitecturas para Desarrollo de Aplicaciones Compuestas con Android, las preguntas son:

1) ¿Considera usted posible desarrollar aplicaciones compuestas para dispositivos móviles con Android sin utilizar una arquitectura de software?
2) ¿Si usted decidiera desarrollar una aplicación compuesta para dispositivos móviles con Android utilizaría esta arquitectura?
3) ¿Le gusta esta arquitectura para apoyar el proceso de desarrollo de aplicaciones compuestas sobre Android en la Universidad de las Ciencias Informáticas?
4) ¿Qué elementos considera positivos de esta arquitectura?
5) ¿Cuáles son sus sugerencias para el desarrollo e implementación de esta arquitectura?

Como parte del trabajo de tesis de maestría del Ing. Orestes Febles Díaz, se realizó un programa informático que permite de forma dinámica y sencilla la aplicación del método de Iadov. Al utilizar esta aplicación con los datos obtenidos de un grupo de 15 expertos, se obtuvieron los siguientes resultados:

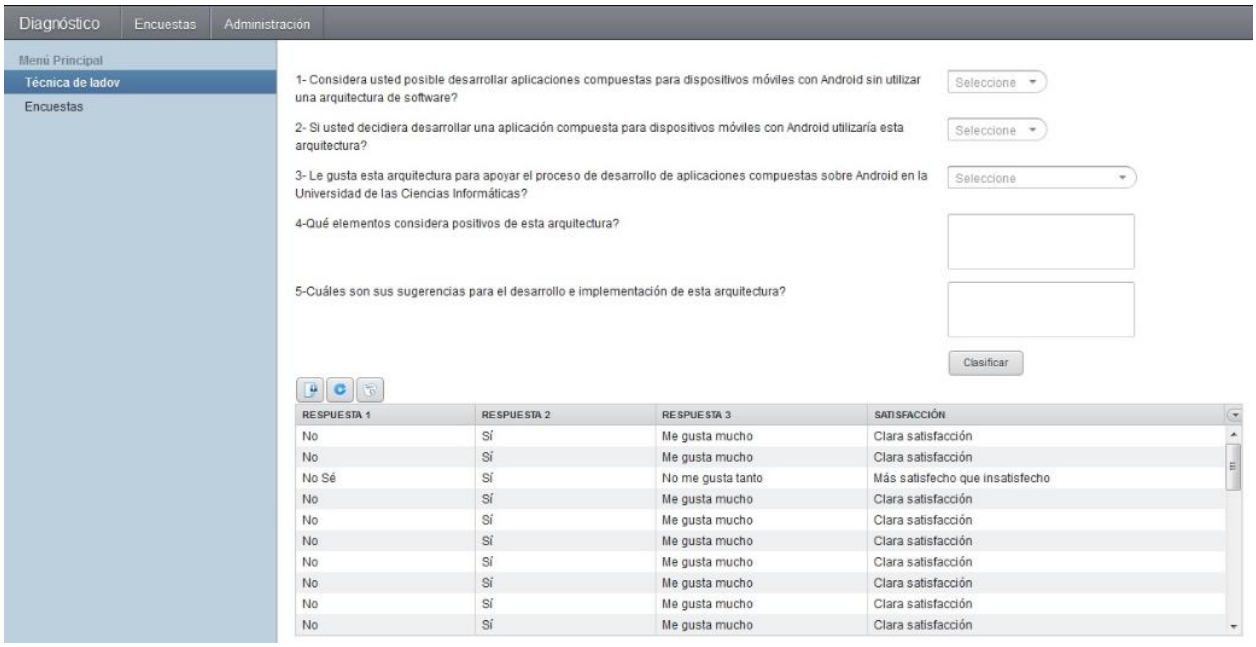

Figura 12 – Programa informático para la aplicación de la Técnica de Iadov

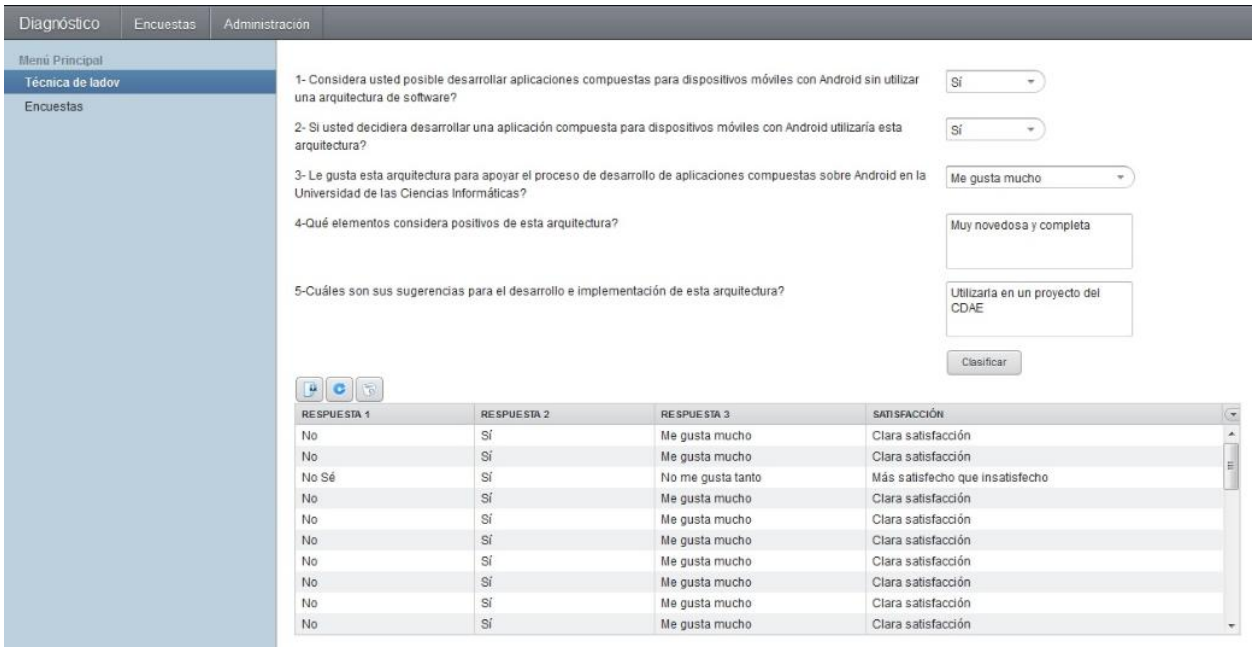

Figura 13 - Respuestas de uno de los expertos

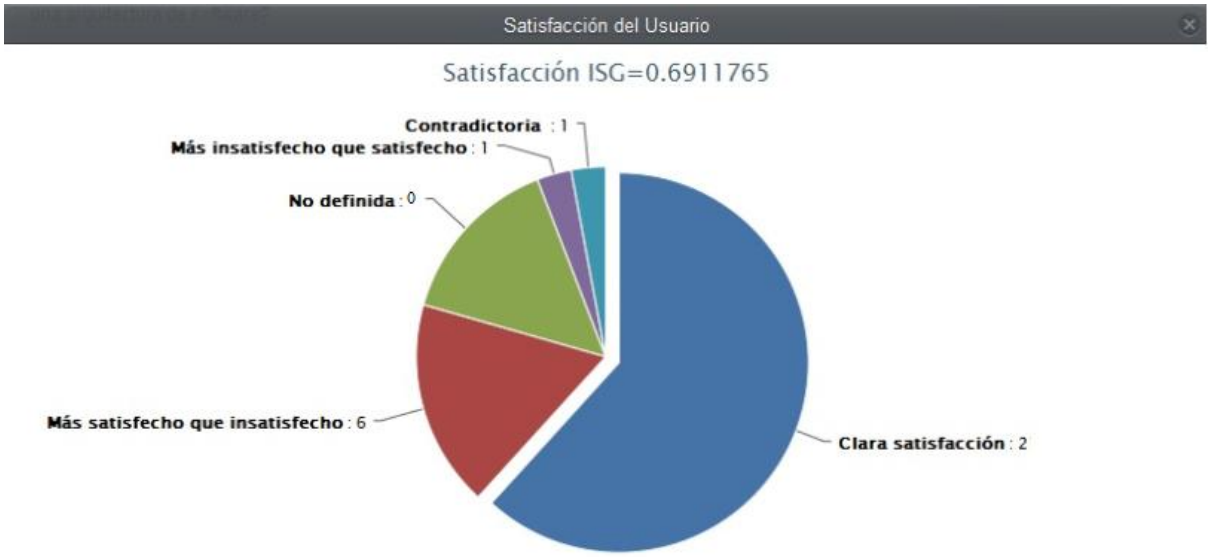

Figura 14 - Índice de Satisfacción Grupal de los expertos

3.3 Validación mediante la implementación de un escenario real de la Universidad de las Ciencias Informáticas

Esta etapa vendrá marcada por la elección del lenguaje, plataforma y herramientas de desarrollo, teniendo en cuenta tipo de dispositivo con el que se va a trabajar, se deciden la configuración y perfil más adecuados.

Los pasos a seguir en esta fase hasta instalar el programa en el dispositivo serían los siguientes:

- Escritura del código.
- Compilación de la aplicación.
- Eliminación de información de clases innecesaria. Esta etapa es opcional y en ella se renombran clases, métodos, interfaces, con el objetivo de evitar ambigüedades. Además, reduce el tamaño de los ficheros de clase, dando lugar a ficheros APK más pequeños.
- Ejecución en un emulador apropiado.
- Instalación en el dispositivo y ejecución. En este caso existen dos modos de hacerlo: en el primero, se descargará la aplicación a través de una conexión de red, se cargará en memoria, se ejecutará la aplicación, y finalmente se eliminará cualquier

traza de ésta en el dispositivo; en la segunda, y siempre que el dispositivo lo permita, se instalará físicamente.

3.3.1 Caso de estudio práctico sobre un escenario real en la Universidad de las Ciencias Informáticas

La práctica de deportes contribuye al buen estado de salud, mejora el funcionamiento general del organismo humano y eleva la calidad de vida. En la UCI, los estudiantes con las aptitudes necesarias pueden además de ejercitarse individualmente, integrar los equipos universitarios de balonmano, básquet, beisbol, futbol 11, futbol sala y voleibol. Estos equipos no entrenan esporádicamente, mantienen un sistema regular de tres encuentros semanales; y de cuatro encuentros en época de competencias, cada uno dura entre una y dos horas. Para los entrenadores resulta difícil determinar en nivel de complejidad del entrenamiento, puesto que es la carga de ejercicios a realizar y la duración del entrenamiento debe estar acorde con las características de los jugadores y las posibilidades reales de alimentación de los mismos. Ante este problema se desarrolló una aplicación que permite:

- ✓ Determinar para cada integrante de cada equipo deportivo, la carga aceptable para el entrenamiento diario; teniendo en cuenta la edad, la estatura, el peso, el género y las calorías consumidas ese día hasta el momento de entrenar.
- ✓ Determinar si el estado climático de ese día permitirá la realización del entrenamiento.
- ✓ Notificar a los deportistas sobre las características del entrenamiento correspondiente, mediante el envío de correos electrónicos
- ✓ Mostrar el menú diario.
- ✓ Mostrar el pronóstico climático diario.
- ✓ Ver la sección de noticias deportivas del portal web de la FEU.

Con este sistema llamado SportDroid se automatizan los procesos del cálculo de las calorías necesarias para cada deportista y del cálculo de las calorías aportadas por el menú diario; contribuyendo a la toma de decisiones de los entrenadores sobre la práctica correspondiente a cada día, pues se muestran en una vista integrada los datos personales del integrante del equipo, los ejercicios recomendados (el tipo de ejercicio, el tiempo de duración y el número de repeticiones) y el pronóstico del clima para el horario en que se realizan los encuentros.

3.3.2 Diagrama de clases del diseño del sistema

En el diagrama de clases del diseño correspondiente a la aplicación SportDroid se aprecian las relaciones entre ellas, los atributos y operaciones que definen el comportamiento de los objetos dentro del sistema, véase el Anexo 2.

Existe una clase Global, donde se generan variables que podrán ser utilizadas en cualquier momento por todas las restantes clases; en dichas variables se almacenan desde el momento de creación de la aplicación datos esenciales como el listado de los deportistas, los platos del menú del comedor y el estado climático para todo el día.

También hay una interfaz principal, que permite acceder a otras cuatro interfaces y al cliente de correo propio del dispositivo donde se encuentre instalado SportDroid. El contenido mostrado en dos de las vistas, es generado a partir de adaptadores. En las clases CalcularCaloriasMenu y ClimaPronostico, se encuentra la mayor parte de la lógica de negocio.

3.3.3 Diagrama de secuencia del sistema

A continuación se muestra el diagrama de secuencia del sistema SportDroid:

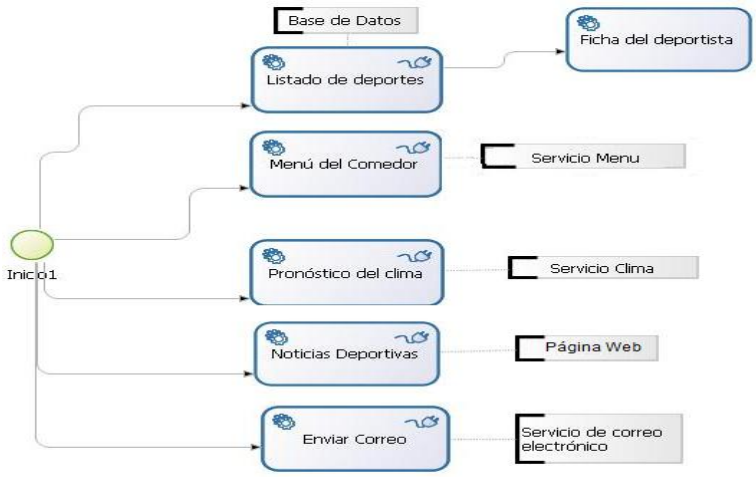

Figura 15 - Diagrama de secuencia de SportDroid

El proceso de navegación dentro de la aplicación comienza en una vista principal, desde la cual se puede pasar a otras 4 vistas o al cliente de correo del dispositivo. El primer icono permite ver un listado desplegable que contiene todos los equipos deportivos con sus integrantes, a partir de ahí el usuario puede seleccionar cualquiera de los deportistas y entonces se muestra la ficha personal correspondiente. En esta ficha aparecen datos personales, así como una serie de ejercicios recomendados para el entrenamiento de ese

día, el pronóstico del clima para el horario de la tarde y facilita el envío de un correo electrónico para informar al jugador la decisión final del entrenador sobre las características de la práctica para ese día.

Los datos personales de cada integrante de los equipos deportivos, son obtenidos de una base de datos hecha sobre el lenguaje SQLite, dentro de una clase perteneciente a la aplicación.

Desde el segundo icono se accede directamente al menú del comedor, desde el tercer icono al pronóstico climático, desde el cuarto icono a la página web de la sección de noticias deportivas del sitio de la FEU en la UCI, y desde el quinto icono al cliente de correo.

La información de los platos disponibles en el menú y del estado del clima para cada sesión diaria, se obtienen del consumo de dos servicios.

La combinación de los datos de los deportistas y de las calorías aportadas por el almuerzo y el desayuno correspondientes a ese día, permiten determinar los ejercicios a realizar.

3.4 Vistas de la aplicación SportDroid

Figura 16 - Pantalla inicial de SportDroid

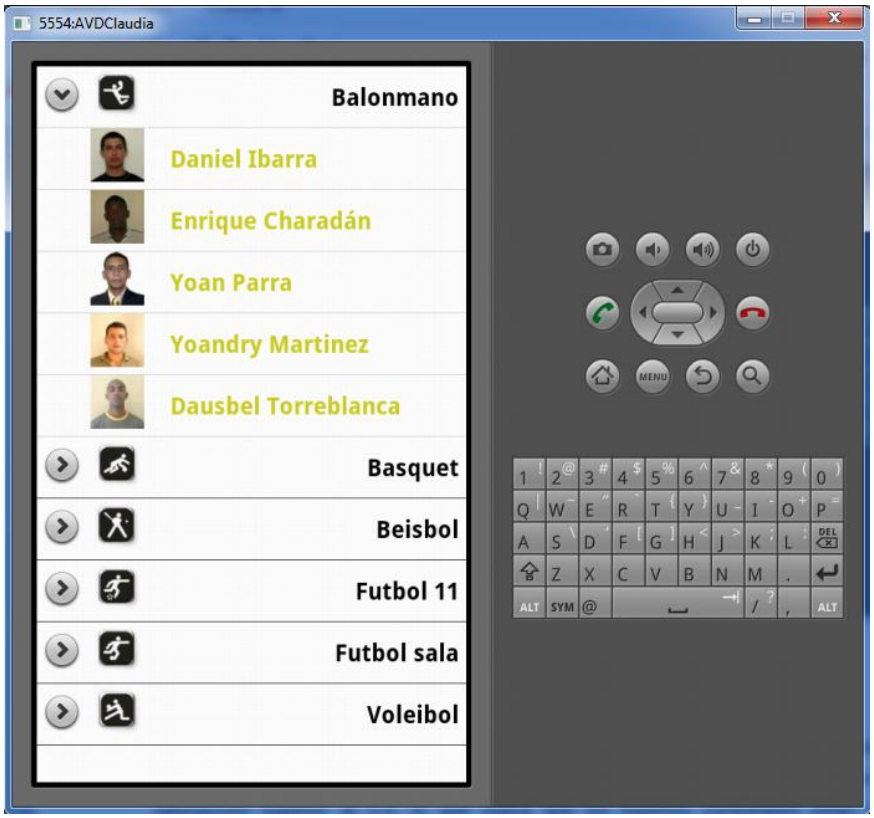

Figura 17 - Listado de los equipos deportivos con sus integrantes

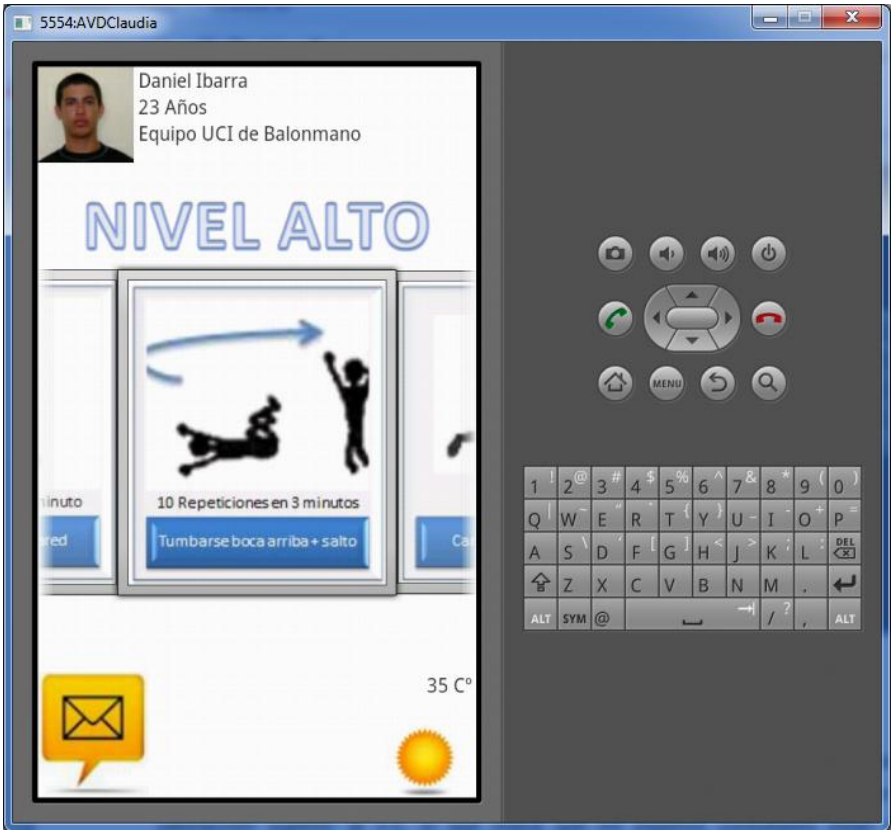

Figura 18 - Ficha de uno de los deportistas

Figura 19 - Menú del comedor

Figura 20 - Pronóstico climático

3.5 Conclusiones parciales

En este capítulo se validó por dos vías la propuesta arquitectónica. Mediante el método de Iadov, utilizado para evaluar cualitativamente la arquitectura, se obtuvo un índice de satisfacción superior al 0.5, lo que indica que los expertos consultados están satisfechos con la solución presentada. Además con el sistema SportDroid, se demostró que los componentes arquitectónicos funcionan correctamente y se integran unos con otros, dando como resultado un funcionamiento eficiente para las aplicaciones compuestas que sean desarrolladas a partir de la arquitectura propuesta en este trabajo.

Conclusiones

Finalizado el trabajo se puede concluir que:

- Se demostró la superioridad de Android como sistema operativo con excelentes posibilidades para permitir el desarrollo de aplicaciones en la Universidad de las Ciencias Informáticas. Se analizaron las características de las aplicaciones compuestas en una arquitectura orientada a servicios y los aspectos necesarios para definir una arquitectura de software.
- Se completó una propuesta arquitectónica para el desarrollo de aplicaciones compuestas para dispositivos móviles con el sistema operativo Android, que incluye varios aspectos que facilitarán el proceso de construcción de software a los equipos de trabajo que utilicen esta plataforma.
- Se validó cualitativamente la propuesta arquitectónica presentada mediante el criterio de un grupo de expertos, utilizando la Técnica de Iadov.
- Se implementó un caso de estudio práctico sobre un escenario real para aplicar la propuesta arquitectónica, validándose su completa funcionalidad y demostrándose la integración plena entre los componentes.

Recomendaciones

Con esta propuesta arquitectónica se facilita considerablemente el desarrollo de aplicaciones compuestas para dispositivos con el sistema operativo Android, por ello se recomienda aplicar esta tecnología en proyectos reales. Sería muy positivo para Cuba, que la empresa ETECSA, utilizara este trabajo como base para implementar algunos paquetes de aplicaciones que puedan ser comercializados libremente y aportar al país avances desde el punto de vista económico y tecnológico.

Bibliografía

1. Echeverria, M., *Apple, líder en mercado de los smartphones.* El Financiaro, 2012(Negocios).
2. dpa, F. *Google: 850.000 dispositivos con Android se activan diariamente* El País.cr, 2012.
3. Digital, D., *Android llega a las 450.000 aplicaciones* La Nación.com.py, 2012.
4. Acuña, C.J., *Arquitecturas de Software*, 2004, Universidad Rey Juan Carlos: España.
5. Ortigosa, A., *Interfaces de Usuario Avanzadas-Arquitecturas de software*, 2008, E.P.S., Universidad Autónoma de Madrid.
6. Naranjo, M., *Fundamentos de Definición de Arquitectura de Software*, 2009: Bogotá, Colombia.
7. Banerjee, A., *Arquitectura móvil.* The Architecture Journal, 2011. **14**: p. 43.
8. Sammons, J., *The Basics of Digital Forensics.* ScienceDirect, 2012: p. 145–162.
9. box, C.W.c.i.a.
10. *Que son los FrameWorks*. The SOA agenda. Soluciones Java, SOA, y BPM, 2011.
11. Yorio, I.D., *Identificación y Clasificación de Patrones en el Diseño de Aplicaciones Móviles.*, in *Facultad de Informática*2010, Universidad Nacional de La Plata.
12. González, I.C.R., *SVES. Desarrollo de Aplicaciones para Dispositivos Móviles*, in *Departamento de Ciencias de la Computación*2003: MÉXICO, D.F.
13. ArcStream Solutions, I., *A Primer on Mobile Application Security*, 2001, White Paper ArcStream Solutions.
14. Ahmad, S.S. *What is a Reference Architecture?* Anything Enterprise Architecture: Social Network for enterprise architecture practitioners, 2008.

15. Eltes, J., *The Reference Architecture - a foundation for successful projects*, 2010, Callista Enterprise AB.
16. Madrid, U.C.d. *La arquitectura J2ME*. 2008.
17. Ferrás, C.B.L. *Todos los caminos conducen a Android*. Blog de desarrollo del Centro de Consultoría y Desarrollo de Arquitecturas Empresariales, 2011.
18. RIM, D.O.d., *BlackBerry Java Development Environment: Guía de conceptos básicos*.
19. Inc., D.O.d.A., *iOS Technology Overview*, 2011.
20. The Microsoft Technologies Company, P.C., *Programar en Silverlight para Windows Phone*, Geeks.ms, Editor 2011.
21. Osorio, J.I., *Aplicaciones Tradicionales vs Aplicaciones Compuestas.*, in *Blog del CDAE >> Desarrollo SOA*2010, UCI: Ciudad de la Habana.
22. Javier, M.S.P.H. *Aplicaciones compuestas con Silverlight/WPF parte I*. Blog de WordPress.com, 2011.
23. Osorio, J.I., *SOA. Una pequeña descripción. Su enfoque en la UCI.*, in *Blog del CDAE >> Desarrollo SOA*, UCI, Editor 2011: Ciudad de la Habana.
24. neoplatform. *Aplicaciones compuestas*. SOA 2007; Available from: www.neoplatform.es/composite-applications.html.
25. Schmidt, D. *Comparación: iOS 5 vs Android 4.0 vs Blackberry OS 7 vs Windows Phone 7.5 Mango*. Tecno-Byte, 2011.
26. A., E.d.T.d.C.S. *ETECSA*. 2012; Available from: http://www.etecsa.cu/?page=telefonia_movil.
27. Vilchez, A. *Que es Android: Características y Aplicaciones*. 2009; Available from: http://www.configurarequipos.com/doc1107.html.
28. Maldonado, D.M. *Utilizando Framework de programación en las Empresas*. 2008; Available from:

http://www.aplicacionesempresariales.com/utilizando-framework-de-programacion-en-las-empresas.html.

29. Pirola, A., *Appcelerator Titanium*, 2010.
30. Inc., A.S. *How PhoneGap Works*. 2012; Available from: http://phonegap.com/about.
31. Rhomobile, I. *Rhodes, mobilize your enterprise apps*. 2012; Available from: http://www.rhomobile.com/products/rhodes/.
32. Alcarazo, Á.B., *Look!: framework para aplicaciones de realidad aumentada en Android*, in *Facultad de Informática*2011, Universidad Complutense.
33. source, S.s.e.s.l.o. *Aplicaciones para Programación | Soluciones en Software Libre y Open Source*. 2012; Available from: http://solcr.org/category/categorias-de-aplicaciones/aplicaciones-para-programacion.
34. *Instalando Eclipse y ADT*. 2010; Available from: http://android.scenebeta.com/tutorial/instalando-eclipse-y-adt.
35. Peláez, J., *Arquitectura basada en capas.*, in *Algunas notas sobre Tecnologias Microsoft*2009.
36. Androideity. *Arquitectura de Android*. 2011; Available from: http://androideity.com/2011/07/04/arquitectura-de-android/.
37. SQLite. *SQLite Welcome*. Available from: http://www.sqlite.org/.
38. Symantec. *Conceptos básicos SSL*. 2011; Available from: http://www.certsuperior.com/ConceptosBasicosSSL.aspx.
39. *Guía Breve de Tecnologías XML*. Available from: http://www.w3c.es/Divulgacion/GuiasBreves/TecnologiasXML.
40. Gracia, J. *Patrones de diseño. Diseño de Software Orientado a Objetos.* 2007 27 de Mayo; Available from: http://www.ingenierosoftware.com/analisisydiseno/patrones-diseno.php.

41. Inc., G. *Introducción a IMAP y POP*. 2012; Available from: http://support.google.com/mail/bin/static.py?hl=es&ts=1668960&page=ts.cs&rd=1.
42. Android, U. *AndFTP cliente FTP para Android*. 2009; Available from: http://www.universoandroid.com/2009/03/27/andftp-cliente-ftp-para-android/.
43. Corp., H., *Soporte*, 2012.
44. *Los diccionarios y las enciclopedias sobre el Académico*, 2010.
45. García, C.G., *DESARROLLO DE APLICACIONES EN ANDROID*, in *Departamento de Informática y Automática*2010, Universidad de Salamanca.
46. Erl, T., *SOA: principles of service design*, P. Hall, Editor 2007.
47. Center, A.R. *Rendimiento de aplicaciones móviles*. Available from: http://resources.arcgis.com/es/content/enterprisegis/10.0/mobile_performance.
48. J.D. Meier, A.H., David Hill, Jason Taylor, Prashant Bansode, Lonnie Wall, Rob Boucher Jr, Akshay Bogawat. *patterns & practices: Application Architecture Guide 2.0* 2008; Available from: http://apparchguide.codeplex.com/wikipage?title=Chapter%2019%20-%20Mobile%20Applications&ProjectName=apparchguide.
49. Lorenzo, L.J.C.T. *Niveles de satisfacción e insatisfacción escolar por las Actividades en el Medio Natural en la Educación Secundaria Obligatoria y el Bachillerato. Aplicación de la técnica Iadov*. 2005; Available from: http://www.efdeportes.com/efd85/iadov.htm.

Anexos

Anexo 1

Historia del sistema operativo Android.

Anexo 2

Diagrama de clases del diseño del sistema SportDroid