Universidad de las Ciencias Informáticas

Facultad 1

“Diseño de una aplicación móvil Android para la reservación de la Transportación Nacional en la Universidad de las Ciencias Informáticas”

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas.

Autora:

Eliani Brown Hernández

Tutor:

Ing. Vladimir Campos Kindelan

La Habana, septiembre 2020.

“Año 62 de la Revolución

Pensamiento

"Todo crecimiento depende de la actividad. No hay desarrollo físico o intelectual sin esfuerzo, y el esfuerzo significa trabajo".

Calvin Coolidge

Declaración de autoría

Yo, Eliani Brown Hernández declaro ser el único autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Vladimir Campos Kindelan

_____

Firma del Tutor

Eliani Brown Hernández

_____

Firma del Autor

Datos de contactos

Autora: Eliani Brown Hernández
Correo electrónico: ebrown@estudiantes.uci.cu

Tutor: Ing. Vladimir Campos Kindelan
Correo electrónico: vladimirc@uci.cu

Agradecimientos

Quiero agradecerles:

Primeramente a mi mamá, mi abuela, mi abuelo y a mi tía, gracias a ustedes soy quien soy. A mi familia en general, y en especial a mi prima Yanelis por todo el apoyo y el aliento recibido en el trayecto de toda mi carrera y aún mayor en este tramo final de la tesis.

A mi tutor, por ayudarme siempre en mis momentos de desesperación pese a su carga de trabajo, por el apoyo, orientación, confianza y paciencia dedicada en el trayecto de la tesis.

Dicen que los buenos amigos en la universidad se vuelven familia y en eso se convirtieron ustedes, mi Lalito y mi Flachis (te advierto no voy a decir eso que has esperado con ansias). Nuestras historias y locuras son interminables, las quiero de corazón, y sepan que sin la compañía y la ayuda incondicional de ambas, estos cinco años vividos en la universidad jamás hubiesen sido tan especiales.

May, Kare, Daniela, Mariam, ustedes y las dos locas de arriba; siento que voy a explotar de la emoción… todos los momentos que vivimos me vienen a la cabeza, risas, llantos, estrés infinito, fiestas, borracheras, y hasta sentimientos inventados jjjjj. Gracias por ser esas personitas tan bellas, ojalá la vida nos siga brindando la oportunidad de revivir momentos como esos. Las amo.

A mi novio Damián, Tatiii llegaste casi de último, pero que te puedo decir… además de la tesis, fuiste tú lo más importante y especial en mi último año de carrera. Les agradezco a ti y Aita por todo el cariño y apoyo incondicional brindado en esta recta final, no imaginan lo importante que han sido para mí. Deseo de todo corazón que pronto seas tú quien este presentando agradecimientos como estos. Gracias por ser parte de mi vida.

Y a todos esas lindas amistades con las que pase momentos hermosos y que de una forma u otra fueron de gran ayuda e hicieron que mi sueño fuera posible.

A todos, muchísimas gracias.

Dedicatoria

A mi mamá, mi abuela y mi abuelo ya que ellos fueron quienes me encaminaron a ser la ingeniera en que me he convertido hoy, transmitiendo y ofreciéndome buenas vibras para seguir adelante y no me dejaron rendir jamás sepan que sin ustedes nunca lo hubiese logrado.
A mis hermanitas que las amo con todo mi ser; les deseo que se formen y estudien para que logren alcanzar todas sus metas y lleguen hasta donde llegué yo y mucho más.

RESUMEN

El presente trabajo de investigación científica consiste en el análisis y diseño de una aplicación móvil para sistema operativo *Android*, que permita realizar la reservación de la transportación nacional en la Universidad de las Ciencias Informáticas (UCI). Para su confección se tuvo en cuenta el estudio de las aplicaciones móviles que manejan servicios de reserva, esencialmente de transporte, permitiéndose identificar los elementos y características distintivas de este tipo de aplicaciones. Igualmente fueron definidas tecnologías y herramientas idóneas para la realización de la futura aplicación. El proceso propuesto es guiado por la metodología de desarrollo de *software Scrum* y como Entorno Integrado de Desarrollo (*IDE* por sus siglas en inglés) *Android Studio*. Además se propone un plan de pruebas de *software* para verificar el correcto funcionamiento de la aplicación una vez realizada. Al concluir este trabajo se obtiene una documentación detallada acerca del desarrollo de aplicaciones móviles, posibilitando la posterior implementación de una aplicación móvil *Android* capaz de mejorar el servicio de reservación que ofrece el Sistema de Gestión de la Transportación Nacional UCI.

Palabras claves: Sistema de Gestión de Transportación, reservación, dispositivos móviles, aplicaciones móviles, sistema operativo *Android.*

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Las Tecnologías de la Información y las Comunicaciones (TIC), han transformado la forma de vivir en sociedad. Estas amplían considerablemente las capacidades físicas y mentales de las personas, así como las posibilidades de desarrollo social. El auge tecnológico ha permitido un auténtico desenvolvimiento en el desarrollo de los dispositivos móviles, que desde sus inicios en la década de los 90 han evolucionado enormemente. A nivel global, el manejo de estos dispositivos y la información que generan se ha convertido en el centro de la toma de decisiones, y su correcta utilización es capaz de definir en la actualidad una posición importante en el ámbito competitivo de la industria, el mercado y los servicios(1).

Hoy en día el acceso a Internet por medio de redes inalámbricas es indispensable en la mayoría de instituciones, hogares y centros de trabajo debido a la masiva utilización de dispositivos portátiles. A partir de los avances tecnológicos y la accesibilidad que tiene casi toda persona para manipular un dispositivo móvil, muchos sistemas informáticos han dado un salto hacia las plataformas móviles que mediante una aplicación permiten hacer las mismas operaciones que se puede realizar en un computador de escritorio, pero con la ventaja de que permiten realizar tareas mucho más complejas en tiempos más cortos y el lugar no es un limitante de su utilización(2).

Muchas de las empresas que se benefician con el uso de las TIC, utilizan las reservas como un potente motor que les permite una mejor planificación y mayor comercialización de sus servicios, productos y personal. La reserva es definida como: la guarda o custodia que se hace de algo con la intención de que sirva a su tiempo(3). Estas empresas utilizan las reservas con el fin de lograr el acceso a sus productos y servicios, durante un tiempo limitado, con antelación a que estos sean brindados. Empresas que pusieron en marcha los primeros sistemas de reservaciones a nivel mundial fueron: Amadeus y Sabré(4).

En nuestro país, diferentes compañías e instituciones han puesto en práctica a partir sistemas informáticos este tipo de reserva, ya sea transporte, alimentos o lugares de trabajo. Un ejemplo de ello lo constituye la Universidad de las Ciencias Informáticas (UCI), universidad certificada e innovadora; que forma profesionales en la rama de la Informática. Dicho centro se encuentra inmerso dentro del proceso de desarrollo tecnológico de Cuba, y tiene entre sus principales objetivos producir *software* y servicios informáticos, a partir de la vinculación estudio – trabajo como modelo de formación(5).

Un precepto de esta Universidad, era que debía contar con una representación de estudiantes de todos los municipios de la isla, con el objetivo de que ningún lugar del país se privara del privilegio de contar con especialistas de las ciencias informáticas comprometidos con la Revolución y la tarea de informatizar la sociedad cubana. El número de estudiantes, trabajadores y profesores de la institución fue ascendiendo a medida que las necesidades del país así lo requerían. Pronto el centro contó en su residencia con un elevado número de personas procedentes de todos los municipios y provincias de Cuba. Actualmente, dispone de alrededor de 4500 trabajadores y docentes de diversas áreas, así como más de 3.000 estudiantes aproximadamente, por lo que la UCI, realizando un esfuerzo reconocible, cada año garantiza la transportación del 60 % de todo ese personal, desde y hacia sus provincias de origen en los períodos vacacionales y de fin de año (6).

Con el desarrollo de las tecnologías, a las personas se le crean nuevas inquietudes y necesidades, es por ello que en dicha entidad se utilizan los teléfonos inteligentes y tabletas electrónicas por parte de los trabajadores y estudiantes, que les permite realizar labores básicas a través de sistemas *responsive*[1] desarrollados en la institución, como: Akademos, Directorio, Intranet, Sistema de Alimentación, entre otros a los cuales solo se puede acceder a sus servicios por la red UCI. Actualmente, la reservación de transportación en la universidad se realiza a través del Sistema de Gestión de Transportación Nacional UCI. La informatización del sistema antes mencionado fue desarrollada por la Dirección de Informatización (DIN), dirección en la cual se está evaluando dicho proceso, y una de las políticas que se planea es que la reservación se realice de manera más organizada y viablemente posible para el usuario. A partir de pruebas realizadas mediante el acceso al sistema a través un computador y un dispositivo móvil, se pudo percibir que si en la universidad se presentan fallas; como interrupciones en la red cableada o dificultades con la corriente eléctrica (lo que ocurre en reiteradas ocasiones) durante la fecha límite para reservar, el usuario no podría realizar la tarea requerida a tiempo y tendría que dirigirse hacia la Dirección de Transportación. Además, al realizar la reservación el sistema no cuenta con una funcionalidad para que el usuario automáticamente lleve consigo la evidencia de la última reservación realizada, lo que podría provocar desconocimiento de detalles de la misma y desorientación en cuanto al bloque de ida, de regreso, fechas u otros datos que el usuario debe conocer.

[1] *Responsive: es una filosofía de diseño y desarrollo cuyo objetivo es adaptar la apariencia de las páginas web al dispositivo que se esté utilizando para visitarlas.*

A partir de la problemática antes mencionada se define como **problema de investigación:** ¿Cómo mejorar la realización de reservación de la Transportación Nacional UCI? Dicho problema está enmarcado en el **objeto de estudio:** desarrollo de aplicaciones móviles para la reservación de servicios; centrándose en el **campo de acción:** implementación de aplicaciones móviles basadas en distribuciones de Linux para la reservación de transporte.

Según lo planteado anteriormente se busca dar cumplimiento al siguiente **objetivo general:** Diseñar una aplicación móvil *Android* que le permita a los usuarios UCI realizar la reservación de la Transportación Nacional en la universidad.

Para darle cumplimiento al objetivo general planteado se responderán las siguientes **preguntas científicas:**

- ¿Cuáles son los fundamentos teóricos que sustentan la implementación de aplicaciones móviles *Android*?
- ¿Qué principios presenta el análisis y diseño de aplicaciones móviles para la reserva de transporte?
- ¿Qué concepción debe tener la aplicación móvil *Android* para la realización la reservación de la transportación nacional UCI?

Para darle cumplimiento a las preguntas científicas planteadas se definen las siguientes **tareas de la investigación:**

- Sistematización de los referentes teóricos-metodológicos relacionados con la implementación de aplicaciones móviles *Android*.
- Selección de las herramientas, tecnologías y metodología, necesarias para el desarrollo de la aplicación móvil *Android*.
- Modelación de la solución para guiar el posterior proceso de implementación de la aplicación móvil *Android* para la reservación de la transportación nacional UCI.

Durante la realización de la investigación se han empleado los **métodos de investigación científica teóricos** que a continuación se plantean:

Métodos de nivel teórico:

- **Inductivo deductivo:** a partir de la estructura e información de la base de datos del Sistema de Gestión de la Transportación Nacional UCI y de la información extraída de las encuestas realizadas a algunos estudiantes y trabajadores que poseen dispositivos móviles en la universidad, este método ha sido

empleado para generalizar el conocimiento adquirido y utilizarlo para un posterior desarrollo de la aplicación a través de conceptos básicos e independientes.

- **Analítico sintético:** dicho método se utiliza para seleccionar y resumir los elementos más importantes de las fuentes bibliográficas analizadas sobre el desarrollo de aplicaciones móviles *Android*, para que de esta manera puedan ser identificadas las tecnologías y herramientas que se utilizarán en el desarrollo de la aplicación.
- **Modelación:** se utiliza para crear el proceso donde se elaboran los modelos referentes al ciclo de vida del *software* que ayudan a dar cumplimiento a las tareas de diseño de los procesos involucrados en la posterior implementación de la aplicación.

Métodos de nivel empírico:

- **Entrevista:** método para fundamentar y precisar el problema a resolver. Se utilizada para establecer las necesidades del cliente, identificar los procesos de transporte que se llevan a cabo actualmente en la UCI y las deficiencias que presentan permitiendo obtener la información necesaria respecto al proceso de reservación de la transportación nacional en la universidad.
- **Análisis documental:** se emplea para la revisión bibliográfica, la revisión de las fuentes básicas de información, el estudio de documentos y demás bibliografías que se han consultado para adquirir el conocimiento para una posterior implementación de la aplicación.

La presente investigación está compuesta por una introducción, tres capítulos, conclusiones, recomendaciones, anexos y referencias bibliográficas. Los tres capítulos están desglosados de la siguiente manera:

Capítulo 1: Soluciones homólogas, tecnologías, herramientas y metodologías para el desarrollo de aplicaciones móviles para la reservación de transporte. En este capítulo se abordan los principales conceptos asociados al desarrollo de aplicaciones móviles así como los referentes teóricos de la investigación para la propuesta de solución. Se realiza un análisis de los principales sistemas informáticos desarrollados, encaminados a la reservación fundamentalmente de transportación. Se describen las herramientas, tecnologías definidas y metodología a utilizar en el desarrollo del *software.*

Capítulo 2: Diseño de la aplicación móvil para la reservación de la Transportación Nacional en la Universidad de las Ciencias Informáticas. En este capítulo queda plasmada una descripción de la solución propuesta. Se realiza el levantamiento de los requisitos funcionales y no funcionales del *software*

así como se define la arquitectura. También son descritos los patrones de diseño a aplicar y los artefactos derivados de la metodología de desarrollo de *software* seleccionada.

Capítulo 3: Planificación de pruebas de la aplicación móvil Android para la reservación de la Transportación Nacional en la Universidad de las Ciencias Informáticas. En este capítulo se exponen algunas descripciones de las pruebas pertinentes a realizar una vez desarrollada la aplicación móvil *Androi*d para la reservación de la transportación nacional UCI, dirigidas a verificar su correcto funcionamiento.

Capítulo 1: Soluciones homólogas, tecnologías, herramientas y metodologías para el desarrollo de aplicaciones móviles para la reservación de transporte

En el presente capítulo se abordan los principales conceptos asociados al desarrollo de aplicaciones móviles destinadas a la reservación de servicios. Conjuntamente se efectúa un análisis del estado del arte de aplicaciones móviles con sistema operativo *Android,* orientadas a la reservación, fundamentalmente de transportación. Además se describen las tecnologías, herramientas y la metodología de desarrollo de *software* a utilizar que deberán ser aplicadas posteriormente en el desarrollo de la propuesta de solución.

1.1. Principales conceptos asociados a la investigación

Durante la investigación se identificaron diferentes conceptos necesarios para el desarrollo de la solución propuesta, a continuación estos se detallan.

Aplicación móvil

Una aplicación móvil es un programa diseñado para funcionar solamente en dispositivos móviles, tabletas o cualquier dispositivo inteligente. Pueden ser encontradas en varias plataformas que dependen del fabricante y desarrollador en cuestión (*Android, IOS*, entre otras plataformas). Su distribución se realiza a través de tiendas virtuales o por otras vías, donde pueden ser adquiridas de forma gratuita o a través de pago(7).

Existen tres tipos de aplicaciones, las aplicaciones nativas, las aplicaciones web y las aplicaciones híbridas. A continuación, se hará una descripción de cada una:

- ✓ **Aplicación nativa**

Una aplicación nativa es la que se desarrolla de forma específica para un determinado sistema operativo móvil, llamado *Software Development Kit*[2] o *SDK*. Cada una de las plataformas, *Android, iOS o Windows Phone,* tienen un sistema diferente. Este tipo de aplicaciones corren de forma más eficiente sobre estos dispositivos, al estar sus componentes diseñados de manera específica para este sistema operativo. Cuando se habla de desarrollo móvil casi siempre se refiere a aplicaciones nativas. La principal ventaja con respecto a los otros dos tipos, es la posibilidad de acceder a todas las características del *hardware* del móvil:

[2] *Software Development Kit: conjunto de herramientas de desarrollo de software que le permite al programador o desarrollador de software crear aplicaciones para un sistema concreto.*

cámara, *GPS* (*Global Positioning System*), agenda, dispositivos de almacenamiento y otras muchas. Esto hace que la experiencia del usuario sea mucho más positiva que con otro tipo de aplicaciones(8).

- ✓ **Aplicación web**

Una aplicación web o *web app* es la desarrollada con lenguajes como *HTML*, *JavaScript* y *CSS*. La principal ventaja con respecto a la nativa es la posibilidad de programar independiente del sistema operativo en el que se usará la aplicación. De esta forma se pueden ejecutar en diferentes dispositivos sin tener que crear varias aplicaciones. Las aplicaciones web se ejecutan dentro del propio navegador web del dispositivo a través de una *URL.* Por ejemplo, en *Safari*, si se trata de la plataforma *iOS*, el contenido se adapta a la pantalla y adquiere un aspecto de navegación app. En realidad, la gran diferencia con una aplicación nativa es que no necesita instalación, por lo que no pueden estar visibles en tiendas de aplicaciones, y la promoción y comercialización debe realizarse de forma independiente. De todas formas, se puede crear un acceso directo que sería muy parecido a instalar la aplicación en el dispositivo. Las *apps web* móviles son siempre una buena opción si el objetivo es adaptar la web a formato móvil(9).

- ✓ **Aplicación híbrida**

Una aplicación híbrida es una combinación de las dos anteriores, se podría decir que recoge lo mejor de cada una de ellas. Las apps híbridas se desarrollan con lenguajes propios de la *web app*, es decir, *HTML*, *JavaScript y CSS,* por lo que su uso es permitido en diferentes plataformas, pero también dan la posibilidad de acceder a gran parte de las características del *hardware* del dispositivo. La principal ventaja es que, a pesar de estar desarrolladas con *HTML, Java o CSS*, es posible agrupar los códigos y distribuirlas en las tiendas de aplicaciones. Los otros dos grupos son más utilizados en cuanto al desarrollo de aplicaciones móviles se refiere. Un buen ejemplo de aplicaciones híbridas es *Facebook*(10).

A simple vista podría parecer que las aplicaciones nativas tienen más ventajas; pero lo cierto es que en dependencia del tipo de aplicación que se vaya a desarrollar puede convenir utilizar uno u otro método. Después de analizar las diferentes características observadas anteriormente se recomienda que la propuesta de solución sea una aplicación móvil nativa. Pues constituye la opción viable para lograr resolver la problemática existente ya que estas posibilitan la utilización de todos los elementos y recursos del *hardware* del teléfono. Además ofrece una experiencia de usuario completa al ser el tipo de aplicación más desarrollado a nivel mundial, posee mejor estética, diseño, utilidad, accesibilidad, rendimiento y usabilidad.

Proceso de desarrollo de aplicaciones móviles

Para el desarrollo de aplicaciones móviles se debe tener en consideración cada una de las limitaciones de los dispositivos para los cuales se desarrollan. Estos dispositivos móviles funcionan con batería, poseen varios tamaños de pantalla así como distintas configuraciones.

El proceso de diseño y desarrollo de una aplicación móvil se puede dividir en cinco partes fundamentales las cuales son(11):

- **Conceptualización:** se parte de una idea que permita dar cumplimiento a una necesidad o ayudar a realizar una actividad determinada. Se debe responder a las expectativas reales y realizables, por lo que es imperiosa la necesidad de hacer un análisis de la viabilidad del concepto que se quiere realizar(11).
- **Definición:** se procede a definir las funcionalidades de la aplicación en consonancia con el perfil de los usuarios y las especificaciones técnicas. Se establecen los parámetros de acceso al *hardware* del dispositivo y para qué tipo de mercado específico se va a desarrollar. El dimensionado de todo ello permitirá determinar el alcance del proyecto, su duración, coste económico, complejidad del diseño y programación de la aplicación(11).
- **Diseño:** en esta etapa se desarrollan los aspectos definidos de la etapa anterior. Se realiza primeramente, un diseño representativo sin gráficos que será sometido a pruebas por varios usuarios. Una vez superada esta prueba inicial el diseño final es entregado al desarrollador en forma de archivos y pantallas estructuradas para que pueda ser añadido el código de programación. Los sistemas operativos ofrecen la posibilidad de interactuar con el usuario, presentando en la pantalla los elementos necesarios para ello de forma distinta, lo cual debe tenerse en consideración por parte de los diseñadores, pues el diseño de la aplicación influye de manera significativa tanto en el costo económico de la misma como en su desarrollo(11).
- **Desarrollo:** el programador, en función del tipo de aplicación se encarga de desarrollar lo expuesto en los diseños, crear la estructura sobre la cual se apoyará el funcionamiento de la aplicación y el código funcional mediante un lenguaje de programación. Existen varios lenguajes de programación entre los que destacan: *Action Script, ADA, C, C#, C++, Java, ASP,* entre otros(11).
- **Publicación:** una aplicación se publica después de realizarse una serie de pruebas para comprobar su correcto y estable funcionamiento, sin errores de usabilidad, diseño, cumpliendo las políticas y requerimientos de las tiendas de aplicaciones. Durante toda la vida útil de la aplicación, es necesario

evaluar el comportamiento y desempeño de la aplicación, detectar, corregir errores y realizar mejoras o actualizaciones(11).

Distribuciones de Linux

Una distribución de *GNU/Linux* es una colección de *software* que forma un sistema operativo basado en el núcleo *Linux.* Las mismas utilizan una versión específica del núcleo *Linux*, un formato de empaquetado determinado para gestionar sus aplicaciones, así como un conjunto de aplicaciones, en versiones determinadas, que garantizan el correcto funcionamiento de todo el sistema. *Linux* (o *GNU/LINUX,* más correctamente) es un Sistema Operativo como *MacOS, DOS o Windows,* basado en *Software Libre.* Hay que aclarar que *Linux* es solo el *kernel, GNU* es el conjunto de aplicaciones. Sus distribuciones no son más que la recopilación de un *kernel de Linux* y un conjunto de aplicaciones disponibles en el mundo del *Software Libre*. Ejemplos: *Debian, Red-Hat (Fedora), Mandriva, Suse, Ubuntu, Android*…(12).

Android

Android es un sistema operativo multidispositivo, inicialmente diseñado para teléfonos móviles como los sistemas operativos *iOS (Apple), Firefox OS (Mozilla)* y *BlackBerry OS*. En la actualidad se puede encontrar también en múltiples dispositivos, como ordenadores, tabletas, *GPS*, televisores, discos duros multimedia, mini ordenadores, cámaras de fotos, etcétera. Incluso se ha instalado en microondas y lavadoras. *Android* es una plataforma basada en *Linux*, que es un núcleo de sistema operativo libre, gratuito y multiplataforma(13).

Arquitectura de las aplicaciones móviles Android

Las aplicaciones móviles *Android* presentan una arquitectura formada por cuatro capas, y una de las características más importante es que todas las capas están basadas en S*oftware Libre*. Estas etapas se detallan a continuación(14):

- **Aplicaciones:** las aplicaciones base incluyen el cliente de correo electrónico llamado *Gmail*, programa de *SMS, Google calendar*, *Google Maps*, contactos, entre otros. Todas las aplicaciones están escritas en lenguaje de programación *Java*(14).

- **Marco de trabajo de aplicaciones:** los desarrolladores tienen acceso completo a las mismas *API*[3] del entorno de trabajo usados por las aplicaciones base. La arquitectura está diseñada para simplificar la reutilización de componentes; cualquier aplicación puede publicar sus capacidades y cualquier otra aplicación puede luego hacer uso de esas capacidades (sujeto a reglas de seguridad del *framework*). Este mismo mecanismo permite que los componentes sean reemplazados por el usuario(14).
- **Bibliotecas:** *Android* incluye un conjunto de bibliotecas de *C/C++* usadas por varios componentes del sistema. Estas características se exponen a los desarrolladores a través del marco de trabajo de aplicaciones de *Android.* Algunas son: *System C library* (implementación de biblioteca C estándar), bibliotecas de medios, bibliotecas de gráficos, *3D y SQLite*, entre otras(14).
- ***Runtime de Android:*** *Android* incluye un set de bibliotecas base que proporcionan la mayor parte de las funciones disponibles en las bibliotecas base del lenguaje *Java.* Cada aplicación *Android* corre su propio proceso, con su propia instancia de la máquina virtual *Dalvik. Dalvik* ha sido escrito de forma que un dispositivo puede correr múltiples máquinas virtuales de forma eficiente. La Máquina Virtual está basada en registros y corre clases compiladas por el compilador de *Java,* que han sido transformadas al formato *.dex* por la herramienta incluida *dx.* Desde la versión 5.0 utiliza el *ART*, que compila totalmente al momento de instalación de la aplicación(14).
- ***Núcleo Linux*:** *Android* depende de *Linux* para los servicios base del sistema como seguridad, gestión de memoria, gestión de procesos, pila de red y modelo de controladores. El núcleo también actúa como una capa de abstracción entre el *hardware* y el resto de la pila de *software*(14).

1.2. Aplicaciones móviles Android que permiten la reserva, a nivel internacional y nacional

Para lograr que estas aplicaciones cumplan con las características necesarias para su correcto funcionamiento en los diferentes dispositivos móviles, es fundamental que se realice un estudio relacionado con las diversas aplicaciones que sean utilizadas para la reservación en disímiles empresas. El análisis se va a centrar en el objeto de estudio definido en la investigación, además se tienen en consideración las características que presentan las aplicaciones móviles destinadas a la reservación de transportación

[3] *API*: La interfaz de programación de aplicaciones es un conjunto de subrutinas, funciones y que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

encontradas.

Aplicaciones móviles Android que brindan servicio de reserva

Existen diversas aplicaciones desarrollas para dispositivos móviles con sistema operativo *Android*, encargadas de brindar al usuario una vía para realizar una reserva en un negocio determinado. A continuación expondremos algunos ejemplos de estas a nivel internacional:

Uber

Uber es una aplicación de movilidad compartida que te permite reservar un viaje o dar una vuelta en patinete con solo pulsar un botón. Solicita un trayecto con conductor o un patinete eléctrico, permitiendo estar en marcha en cuestión de minutos. La versión actual de la aplicación y el requerimiento de *Android* varían según el dispositivo(15).

inDriver

Servicio de taxi alternativo que opera en más de 300 ciudades de 31 países alrededor del mundo. En *inDriver* solicitar un taxi es barato. Los pasajeros son los que ofrecen el costo del viaje. El pasajero puede elegir al conductor con la calificación más alta, con la cantidad más grande de viajes realizados o hacer una elección por modelo del auto o tiempo de llegada al lugar de partida. La versión actual de la aplicación y el requerimiento de *Android* varían según el dispositivo(16).

YEGO

YEGO es una aplicación que te permite encontrar, reservar y conducir motos eléctricas en la ciudad de Barcelona. Los vehículos se encuentran distribuidos por distintos puntos de la ciudad, sin punto fijo de retorno y sin necesidad de estaciones de recarga de batería. La versión actual de la aplicación es 3.4.3 y requiere *Android* 5.0 y versiones posteriores(17).

Bolt

Bolt es una aplicación de transporte para solicitar viajes rápidos y accesibles. *Bolt* está disponible en más de 30 países y 100 ciudades alrededor del mundo. Su versión actual es *CA.5.51* y requiere *Android 4.1* y versiones posteriores(18).

A nivel nacional se tienen:

Viajeros

"Viajeros", aplicación que tiene como objetivo principal informar sobre los viajes nacionales disponibles referentes a los medios de transportes que comercializa la empresa "Servicios de Reservación Viajero" o mejor conocida como "Viajero", los cuales vendrían siendo 3, los ómnibus asociados a la Empresa de Ómnibus Nacionales (EON), los barcos que realizan viajes La Habana-Nueva Gerona y viceversa, dirigidos por la empresa VIAMAR y los trenes pertenecientes a la Unión de Ferrocarriles de Cuba (UFC). La aplicación cuenta con una versión mínima compatible: *Android 3.0 Honeycomb (API level 11)* y versión recomendada: *Android 4.4.2 KitKat (API level 19)*(19).

Viajando

Viajando es una aplicación *(apk*[4]*)* para dispositivos móviles con sistema operativo *Android*, que permite visualizar la disponibilidad de pasajes por ómnibus nacionales que comercializa la empresa Viajeros. Actualmente, fue lanzada una nueva versión de la aplicación en la cual se pone en marcha la comercialización de pasajes entre provincias. La misma funciona en dispositivos móviles con versiones igual o superior a la 4.4.2. Esta versión de la aplicación solo permite consultar la disponibilidad de pasajes. En versiones posteriores se podrá reservar y pagar los pasajes(20).

Trenes

Creada por la agencia de *software* GeoMIX del grupo GEOCUBA, Trenes posee un tamaño de 46.04 MB y está disponible por el momento sólo para dispositivos con sistema operativo *Android*. La *apk* Trenes contiene información completa sobre el nuevo servicio de trenes nacionales incluyendo itinerario de todas las rutas, horarios de salida y de llegada. Incluye también los detalles sobre las paradas que se harán durante el recorrido, así como condiciones generales de todas las modalidades del servicio y los precios de los pasajes(21).

[4] APK (Aplicación empaquetada de Android): es un paquete para el sistema operativo Android. Este formato se usa para distribuir e instalar componentes empaquetados para la plataforma Android para teléfonos inteligentes, tabletas y algunas distribuciones enfocadas a su uso en ordenadores personales de escritorio y portátiles.

En el ámbito nacional no se encontraron publicaciones relacionadas con aplicaciones móviles *Android* encargadas de brindar servicios de reserva de transporte para el sector estatal, esto no determina la importancia del desarrollo del objetivo propuesto, por lo que a partir de un análisis de las aplicaciones móviles *Android* descritas anteriormente, se concluye en que a pesar de que ninguna resuelve el problema de la investigación ya que estas se encuentran enfocadas a un negocio o área específica y no se especifica en la documentación cómo son realizados los procesos de desarrollo de las mismas, estas cumplen con algunas de las características definidas y brindaron información oportuna para un mayor conocimiento acerca del posterior desarrollo de la propuesta de solución, en cuanto a la tendencia de diseño de estas aplicaciones.

1.3. Herramientas utilizadas

Herramienta para el modelado Visual Paradigm 8.0

Es una herramienta *CASE (Computer Aided Software Engineering)*. La misma propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, a través del paso por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. Ha sido concebida para soportar el ciclo de vida completo del proceso de desarrollo del *software* a través de la representación de todo tipo de diagramas(22).

Constituye una herramienta privada disponible en varias ediciones, cada una destinada a satisfacer diferentes necesidades: *Enterprise, Professional, Community, Standard, Modeler* y Personal. Existe una alternativa libre y gratuita de este *software*, la versión *Visual Paradigm for UML 6.4 Community Edition*. Fue diseñado para una amplia gama de usuarios interesados en la construcción de sistemas de *software* de forma fiable a través de la utilización de un enfoque orientado a objetos(22).

Esta herramienta permite aumentar la calidad del *software,* a través de la mejora de la productividad en el desarrollo y mantenimiento del *software*. Aumenta el conocimiento informático de una empresa para ayudar a la búsqueda de soluciones para los requisitos. También permite la reutilización del *software,* portabilidad y estandarización de la documentación, además del uso de las distintas metodologías propias de la Ingeniería de *Software*(22).

Sistema gestor de base de datos PostgreSQL 9.5

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia *BSD* y con su código fuente disponible libremente. Utiliza un modelo cliente/servidor y usa multiprocesos en lugar de multihilos, para garantizar la estabilidad del sistema. De esta forma, un fallo en uno de los procesos no afectará al resto y el sistema continuará funcionando(23).

Este gestor es utilizado para el análisis de la base de datos del Sistema de Gestión de la Transportación Nacional UCI, misma que deberá ser utilizada por la aplicación a desarrollar.

Bases de Datos en Android (SQLite)

SQliteStudio es un programa de código abierto para la gestión de bases de datos *SQLite* que proporciona una interfaz gráfica intuitiva que facilita la creación y consulta de este tipo de bases de datos. Este *software* está disponible para las plataformas más comunes: *Windows, MacOSX y Linux*. Las ventajas de este s*oftware* respecto a otros similares son(24):

- Permite la exportación de datos a distintos formatos *(CSV, HTML, XML, PDF, JSON)* y la importación de datos a través de un fichero *CSV*.
- No requiere instalación para su uso.
- Admite complementos oficiales y de terceros para expandir las funcionalidades. Cabe destacar el complemento *"DbAndroid"*, que permite acceder directamente a bases de datos en el dispositivo *Android*, en lugar de tener que modificar la base de datos en una carpeta local para después enviar esta al dispositivo.

PgAdmin III 1.22.1

PgAdmin III es un completo sistema de gestión y diseño de base de datos *PostgreSQL* para sistemas *Unix* y *Windows.* La misma nos permite gestionar y diseñar nuestra base de datos de manera fácil e intuitiva, es una herramienta de código abierto el cual nos permite su uso sin ningún tipo de costo. Está disponible gratuitamente bajo los términos de la licencia *PostgreSQL* y puede redistribuirse siempre que se cumplan los términos de la licencia(25).

PyCharm

PyCharm se describe como un Entorno de Desarrollo Integrado (*IDE,* por sus siglas en inglés), orientado al desarrollo de *software* en lenguaje *Python,* aporta a los desarrolladores varias características como la integración con un sistema de control de versiones y soporta el desarrollo *web* con *Django*(26).

Entorno de Desarrollo Integrado Android Studio 3.5.3

Android Studio es el entorno de desarrollo integrado *(IDE)* oficial para el desarrollo de aplicaciones *Android* (apps). Se basa en un entorno de desarrollo integrado de *Java* para *software* llamado *IntelliJ IDEA*[5]. El *IDE* realiza las compilaciones de código a través de *Gradle* que es un sistema de compilación que se basa en la máquina virtual de *Java (Java Virtual Machine JVM),* lo que implica que es posible programar un *script* en *java* y el *IDE* lo va a comprender. *Android Studio* utiliza la característica *Instant Push* que permite lanzar cambios de código fuente a la aplicación *Android* en tiempo de ejecución sin necesidad de reiniciar la app. *Android Studio* tiene una arquitectura recomendada, *Retrofit*, que hace uso del principio de diseño *Separation of Concerns SoC* o separación de problemas y del modelo de persistencia que permite(27):

- Mantener los datos en caso de que el sistema operativo *Android* del dispositivo móvil finalice la app con el fin de liberar recursos.
- Garantizar la continuidad del funcionando de la app, cuando haya intermitencia o se pierda la conexión a *Internet*.

Librería Android Room 2.2.0

Android Room es un componente incluido en la librería *Android Jetpack* que proporciona una capa de abstracción sobre *SQLite*, construyendo un acceso a la base de datos más robusta. Comparte el mismo objetivo que las librerías de *Jetpack*, conseguir crear bases de datos de una forma más sencilla, permitiendo la persistencia de los datos en la aplicación incluso sin conexión a *Internet*(28).

Django 3.0.3

Django es un *framework* de desarrollo web de código abierto, escrito en *Python* y que respeta el patrón de diseño conocido como Modelo-Vista-Controlador. La meta de *Django* es la creación de sitios webs complejos de forma ágil y fácil, poniendo énfasis en la reutilización, conectividad y extensibilidad de componentes y su principio no te repitas (*DRY, Don´t Repeat Yourself*)(29).

[5] *IntelliJ IDEA es un IDE para la creación de programas informáticos desarrollado por JetBrains.*

Django REST Framework

Django REST Framework es una librería, que nos permite construir una *API REST* sobre *Django* de forma sencilla. Ofreciendo una alta gama de métodos y funciones para el manejo, definición y control de nuestros recursos. Incluye varios aspectos importantes en el diseño y creación de *APIs* tales como la autenticación. Los conceptos más usados durante la creación de una *API REST* son los siguientes(30):

- *Serializers:* Permite que los datos complejos, como consultas e instancias a tipos de datos nativos de *Python*, puedan ser fácilmente transmitidos en otro tipo de datos como *JSON o XML* y viceversa.
- *Viewsets:* Permite combinar la lógica de un conjunto de controladores relacionados en una sola clase. En el *viewset* se realiza la manipulación de las peticiones *HTTP*, además de crear los diferentes *endpoints* de la *API*.

1.4. Lenguajes utilizados

Lenguaje Python 3.7.2

Python es un lenguaje de programación, interpretado, de alto nivel, orientado a objetos y con semántica dinámica. Sus estructuras de datos de alto nivel, combinada con tipado dinámico, lo hacen muy atractivo para el desarrollo rápido de aplicaciones, así como para su uso como un lenguaje de *script* o lenguaje para conectar componentes existentes. Simple y fácil de aprender la sintaxis de *Python*, hace hincapié en la lectura y por lo tanto reduce el costo de mantenimiento del programa. *Python* soporta módulos y paquetes, lo que fomenta la modularidad del programa y la reutilización de código. El intérprete de *Python* y la extensa librería estándar están disponibles en código fuente o binario sin cargo para todas las plataformas y pueden ser distribuidos libremente(31).

Lenguaje Kotlin 1.3.5

Kotlin es un lenguaje de programación de tipado estático que corre sobre la Máquina Virtual de *Java (JVM)* y que también puede ser compilado a código fuente de *JavaScript*. Un punto a favor de *Kotlin* y de su fácil adaptación es que *Android Studio* tiene una completa integración con él, por lo que un proyecto desarrollado en el lenguaje *Java* puede ser fácilmente convertido a *Kotlin. Kotlin* es un lenguaje que reduce drásticamente la cantidad de código repetitivo, es un lenguaje muy expresivo, lo que se traduce en una menor necesidad de escribir código para realizar tareas que con *Java* serian escritas con más verborrea. *Kotlin* es además, un lenguaje ligero, ya que siguiendo una de las máximas de *Android* consigue crear *Apks* que se puedan ejecutar sin problemas en multitud de dispositivos(14).

1.5. Servicios web

Existen múltiples definiciones sobre lo que son los servicios web, pero en todos los casos, el concepto inherente es el de funcionalidades que se encapsulan y se publican, para ser accedidas por otras aplicaciones mediante una serie de protocolos basados en *XML*(32).

También, sería posible hablar de ellos como un conjunto de aplicaciones o de tecnologías con capacidad para interoperar en la web. Estas aplicaciones o tecnologías intercambian datos entre sí con el objetivo de ofrecer servicios. Son aplicaciones modulares basadas en *Internet* que realizan una tarea empresarial específica y se ajustan a un formato técnico particular. El formato técnico asegura que cada uno de estos servicios es una aplicación que se integrará fácilmente con otros servicios para crear un proceso de negocio completo(33).

1.6. Arquitectura SOA

La arquitectura orientada a servicios (*SOA*, siglas en inglés) no es un concepto fácil de definir. Diversos autores afirman que es un enfoque para diseñar y construir soluciones de negocios; tal que permitan, a las organizaciones, unir sus objetivos con la infraestructura de las Tecnologías de la Información y la Comunicación (TIC). *Microsoft Corporation* define *SOA* como: “una estrategia general de organización de los elementos de *IT*, de forma que una colección abigarrada de sistemas distribuidos y aplicaciones complejas se pueda transformar en una red de recursos integrados, simplificada y sumamente flexible”.

Se aclara también que la manera más habitual de implementarla es a través de servicios web, que no significa que estos servicios requieran una arquitectura *SOA*, y viceversa.

La arquitectura *SOA* permite integrar sistemas y aplicaciones heterogéneas e independientes a través de una metodología que logre una óptima integración, como así también, facilidad de adaptación ante cambios posteriores surgidos de la evolución de cualquier organización(34).

SOAP define tres entidades:

SOAP (Simple Object Access Protocol) es un estándar de *W3C,* que define cómo objetos remotos pueden comunicarse mediante el intercambio de *XML* (Lenguaje de Marcas Extensible); es muy recomendable para entornos formales y donde las funcionalidades de interfaz y datos estén claramente definidas. La idea básica es que en la comunicación hay dos partes (cliente y servidor), una de las cuales (el servidor) presta una serie de servicios que son consumidos por la otra (cliente). *SOAP* es un protocolo de comunicación para intercambiar mensajes entre aplicaciones distribuidas sin tener en cuenta su implementación semántica específica y plataforma programadora(33).

En estos momentos, el estándar de comunicación utilizado en la Universidad de la Ciencias Informáticas es *SOAP*, lo cual nos conduce a llevar a cabo un estudio más a fondo de esta tecnología, que será la base para consumir los servicios brindados por la universidad a través de los diversos dispositivos que corren sobre la plataforma *Android.*

1.7. API REST

Los servicios *Web REST* (Estado de Transferencia Representacional o *Representational State Transfer*) "es una técnica de arquitectura *software* que se apoya totalmente en el estándar *HTTP*. Nos permite crear servicios y aplicaciones que pueden ser usadas por cualquier dispositivo o cliente que entienda *HTTP*. Por lo tanto, *REST*, es el tipo de arquitectura más natural y estándar para crear *APIs* (*Application Programming Interface*)(35).

El objetivo que se persigue, en la construcción de aplicaciones distribuidas e inspiradas en las características de la web, es que en lugar de basar la comunicación en protocolos llamados "pesados" como *SOAP, REST* se basa en el envío de mensajes sobre *HTTP y XML*(34).

Un pedido *HTTP* consta de cuatro métodos de acceso(35):

- *GET*: Para consultar y leer recursos (con el cual se pide un recurso dado, generalmente sin efectos secundarios).
- *POST*: Para crear recursos (realiza un pedido, acceso o modificación de un recurso).
- *PUT*: Para editar recursos (agrega en el mensaje, una representación de un recurso destinada al servidor).
- *DELETE*: Para eliminar recursos (elimina un recurso dado).

Además, se deben usar los códigos de error de *HTTP* para representar el resultado de una llamada a la *API.* Algunos de ellos son los siguientes(35):

- *200 OK:* Petición realizada correctamente.
- *201 Created*: Recurso creado correctamente.
- *400 Bad Request:* Petición fallida.
- *401 Unauthorized:* No autorizado, autenticación requerida.
- 404 *Not found*: Recurso no encontrado.
- 500 *Internal Server Error*: Error interno del servidor.

Las funcionalidades de la aplicación móvil *Android* para la reservación de la transportación nacional UCI

deben ser implementadas a partir de la *API REST*, ya que la *API* es la interfaz entre el Sistema de Transportación Nacional UCI y la apk.

1.8. Metodología de desarrollo de software

Todo proceso de desarrollo de *software* debiera estar guiado por una metodología que permita estructurar, planear, y controlar dicho proceso, además de adoptar buenas prácticas que nos permitan asegurar que el *software* estará construido a conciencia(36).

Dicho concepto incluye modelos de sistemas, notaciones, reglas, sugerencias de diseño y guías de procesos. Existen metodologías tradicionales o robustas, centradas específicamente en el control del proceso, y ágiles, que cambian significativamente algunos de los énfasis de los métodos ingenieriles para lograr un mayor enfoque solamente en el desarrollo del proyecto(37).

Comparación entre modelos:

La siguiente tabla muestra aspectos relevantes de las metodologías de desarrollo tradicional contrastándolas con los aspectos relevantes de las metodologías de desarrollo ágil(38).

Tabla 1. Comparación entre modelos.

Metodologías tradicionales	**Metodologías ágiles**
Predictivos	Adaptativos
Orientados a procesos	Orientados a personas
Proceso rígido	Proceso flexible
Se concibe como un proyecto	Un proyecto es subdividido en varios proyectos más pequeños
Poca comunicación con el cliente	Comunicación constante con el cliente
Entrega de *software* al finalizar el desarrollo	Entregas constantes de *software*
Documentación extensa	Poca documentación

Para el desarrollo del sistema propuesto se realiza un estudio de las metodologías ágiles, puesto que, según las características evidenciadas anteriormente, se concluye en que estas convergen de manera más completa con la problemática a resolver, centrándose en la metodología de desarrollo ágil *Scrum*.

Metodología Scrum

Scrum es un método de desarrollo ágil de *software*. Los principios *Scrum* son congruentes con el manifiesto ágil y se utilizan para guiar actividades de desarrollo dentro de un proceso de análisis que incorpora las siguientes actividades estructurales: requerimientos, análisis, diseño, evolución y entrega. *Scrum* acentúa el uso de un conjunto de patrones de proceso del *software* que han demostrado ser eficaces para proyectos con plazos de entrega muy apretados, requerimientos cambiantes y negocios críticos. Los patrones de proceso *Scrum* permiten que un equipo de *software* trabaje con éxito en un mundo en el que es imposible eliminar la incertidumbre(39).

No existe una metodología de *software* universal. Cada equipo de desarrollo escoge la metodología según las características de su proyecto, por lo que es importante determinar el alcance del proyecto antes de escoger la metodología que se va a emplear en el desarrollo del mismo. *Scrum* es una forma de gestionar un equipo de manera que trabaje de forma eficiente y de tener siempre medidos los progresos, de forma que se sepa por qué etapa se encuentra el proceso(22).

Para la realización de este proyecto se propone elegir *Scrum*, ya que es un método adaptativo de gestión de proyectos que se basa en principios ágiles, constituyendo esta una de las características fundamentales que se tuvo en cuenta en la elección de la metodología. Además, esta metodología es empleada por la DIN actualmente, para el desarrollo de los procesos del Sistema de Transportación. El uso de esta permite la obtención de la lista de especificación de requisitos a implementarse posteriormente, o *Product Backlog*. Posibilita además, la especificación de las funcionalidades del sistema, por iteraciones, con el objetivo de lograr una planificación óptima de tiempo. Se definen fechas en relación con el tiempo disponible, obteniéndose al final de cada iteración un producto potencialmente entregable al cliente, con las mínimas funcionalidades del software, asegurando desde el inicio la buena comunicación y motivación de este. Antes estas razones, se justifica la elección de esta metodología para el desarrollo de la propuesta de solución.

1.9. Conclusiones parciales

El estudio de todos los conceptos expuestos, en conjunto con el análisis al Sistema de Gestión de Reservación de Transportación Nacional UCI, permitió obtener los principales elementos teóricos que sustentan la elaboración de una aplicación móvil para dar solución a la problemática de la investigación. Con el análisis de los sistemas homólogos existentes se comprobó que ninguna de estas aplicaciones da solución al problema planteado pero aportaron información oportuna en cuanto a la tendencia y diseño de

aplicaciones móviles. Sin descartar que el estudio de las herramientas, tecnologías y metodologías de desarrollo de *software*, permitió crear la base tecnológica adecuada para un posterior desarrollo de este tipo de aplicación.

Capítulo 2: Diseño de la aplicación móvil para la reservación de la Transportación Nacional en la Universidad de las Ciencias Informáticas

En el presente capítulo son citados los resultados que se alcanzan en las fases de conceptualización y diseño de la aplicación móvil *Android* para la reservación de la transportación nacional UCI. Una vez realizada una descripción del entorno a través de un modelo de dominio en el que se analizan los conceptos, las entidades y sus relaciones, se realiza la definición de las principales características a las cuales debe dar cumplimiento la aplicación en cuestión de requisitos funcionales y no funcionales. Igualmente se describen las historias de usuarios que posteriormente darán paso a las funcionalidades de la aplicación.

2.1. Descripción del proceso de reservación de la transportación Nacional UCI

El proceso comienza cuando la Rectoría envía el ‘’Plan calendario de transportación’’ donde se especifican las fechas de la transportación masiva para los respectivos pases, entiéndase: pase fin de año y pase fin de curso. Con estas fechas se crea un plan de transportación, el cual es presentado al Ministerio de Transporte (MITRANS). De ser aprobado, es consultado por el MITRANS y se asignan los transportes a la Universidad, en caso contrario se cambian las fechas del plan, se confecciona uno nuevo y se presenta nuevamente (ver anexo 1).

Con los transportes asignados, la Dirección de Transporte habilita el sistema e informa a la comunidad universitaria, que ya es posible realizar la confirmación de su transportación en el período establecido. Mientras el sistema se encuentre activo, los usuarios tienen la posibilidad de acceder a diferentes opciones que este sistema les brinda, para mantenerse informado acerca del proceso de gestión de transportación. Los usuarios UCI tienen la posibilidad de reservar, dígase hacia sus provincias o realizar una permuta; además a estos usuarios también se les permite realizar la reservación a un familiar asociado. En caso de que el usuario no tenga dominio UCI (ejemplo personal XETID), su reservación será realizada a través de un tramitador tercero (ver anexo 1).

Una vez que se cierra el período de confirmación se distribuyen por ómnibus las reservaciones de acuerdo a las rutas antes especificadas y se generan los listados que posteriormente son entregados a los responsables de cada transporte. El responsable después de recibir los listados, verifica el uso de la transportación de cada viajero. Luego de culminado el período de transportación el responsable de transporte entrega el listado a la Dirección de Transporte, que procede a bloquear el acceso al sistema a los usuarios que no usaron la transportación (ver anexo 1).

Para arribar a la solución propuesta descrita en el siguiente epígrafe fueron aplicados diferentes métodos, uno de estos fue la aplicación de una encuesta (ver anexo 2). Esta encuesta tenía como objetivo identificar puntos fundamentales que se deben tener en cuenta en el posterior desarrollo de dicha solución, algunos de estos puntos fueron: el sistema operativo que más predominaba en la UCI y si sería útil desarrollar la investigación. Para realizar la encuesta fue necesario definir la población, unidad de estudio y muestra.

- Población de la investigación: estudiantes y trabajadores de la UCI con dispositivos móviles.
- Unidad de estudio: estudiantes y trabajadores de la Facultad 1 con dispositivos móviles.
- Muestra: 42 trabajadores del Centro de Innovación y Desarrollo de Internet (CIDI) y 329 estudiantes de 1ro, 4to y 5to año conjuntamente de la Facultad 1 con dispositivos móviles, que se encontraban en sus aulas en el momento de realizar la encuesta; estos fueron seleccionados a través de un muestreo intencional.
- Muestreo: para seleccionar la muestra se utilizó la técnica de muestreo intencional, para la selección del centro productivo CIDI y once grupos de dicha facultad.

El total de personas a encuestar era de 401, en el momento de su aplicación se encontraban en su puesto de trabajo 371 estudiantes y trabajadores, por lo que 30 se encontraban ausentes (ver Tabla 18).

Al aplicar la encuesta se identificó que desarrollar una aplicación para realizar la reservación de la transportación nacional UCI desde un dispositivo móvil, podría ser de utilidad a los usuarios de la universidad ya que 351 personas creían necesario el desarrollo de la aplicación y solo 20 dieron respuestas negativas (ver Ilustración 5).

Una vez procesada la encuesta se obtuvieron los siguientes resultados: 335 usuarios con dispositivos con sistema operativo *Android,* 16 con *IOS,* 5 con *Windows Phone* y 15 con otros (ver Tabla 19).

Con la obtención de esta información se concluye que el sistema operativo que predomina es *Android*; teniendo en cuenta esto se decide proponer la realización de la aplicación para que funcionen en este tipo sistema operativo.

2.2. Propuesta de solución

La aplicación móvil asociada al Sistema de Gestión de la Transportación Nacional UCI debe estar orientada a la reservación de la transportación nacional de la universidad, cumpliendo como misión mejorar la realización de dicha tarea. En la misma los usuarios con dominio UCI podrán reservar, cancelar y ver los

detalles tanto de su transportación como la de sus familiares asociados, a los que les podrá reservar de igual forma. La aplicación poseerá sistema operativo *Android* en su versión 4.4 hacia adelante. Además contará con un diseño arquitectónico consistente, intuitivo, agradable para el usuario y adquirirá como ventaja que si el usuario cierra la aplicación e inmediatamente o pasados unos días vuelve a acceder a la misma, podrá observar los datos tal y como se encontraban ya que estos persistirán de manera local. Por tanto, el usuario llevará consigo registrada la última actualización de la reservación realizada. Asimismo una vez realizada la reservación, la aplicación enviará una notificación de "La reservación ha sido realizada satisfactoriamente"

2.3. Modelo de dominio

Su utilidad radica en ser una forma de "inspiración" para el diseño de los objetos *software*. Un modelo de dominio muestra las clases conceptuales significativas en un dominio del problema y se centra en las abstracciones relevantes, vocabulario del dominio e información del dominio. Es el artefacto clave del análisis orientado a objetos(40).

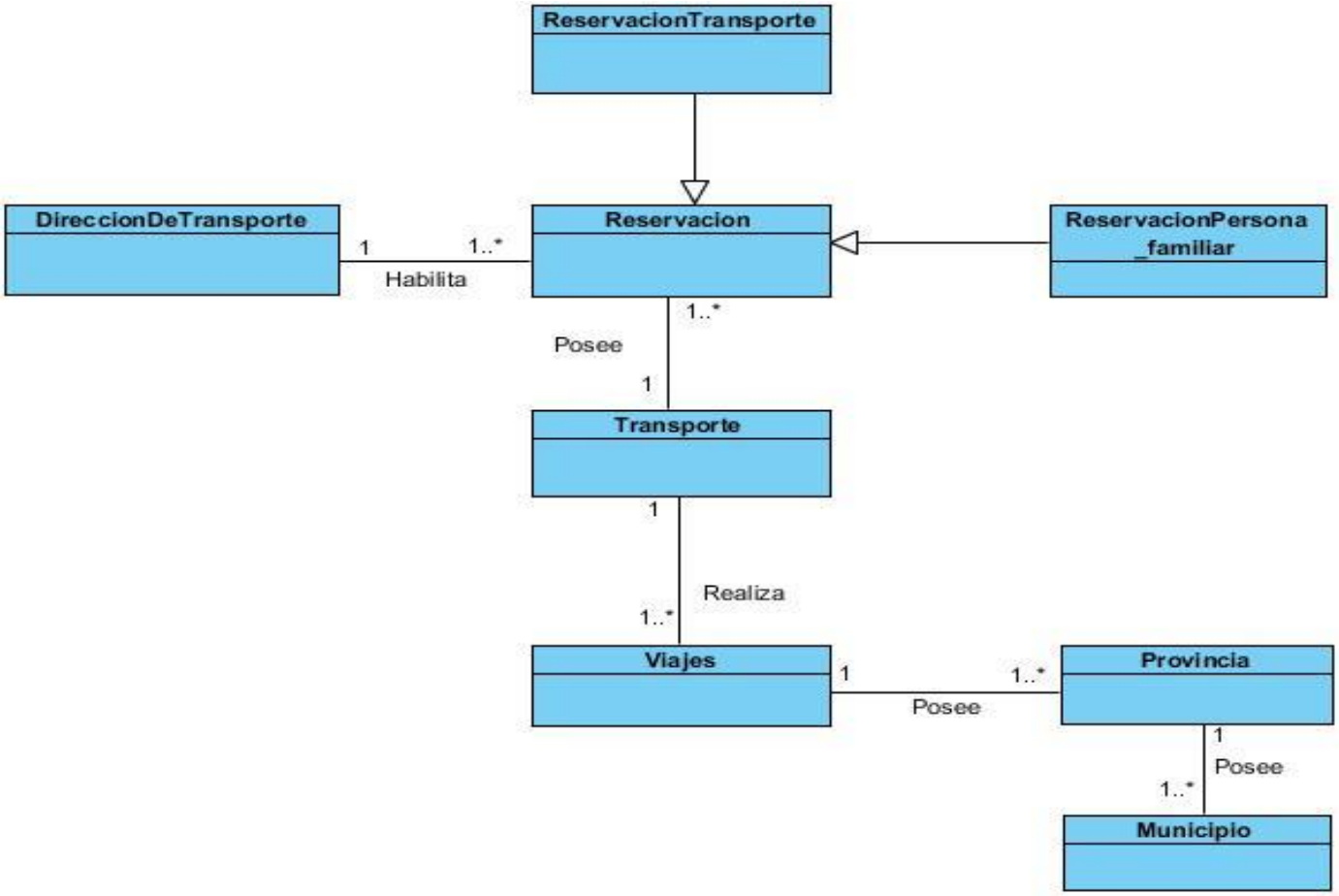

Ilustración 1. Modelo de dominio de la aplicación móvil Android para la reservación de la transportación nacional UCI.

En el anterior modelo de dominio se observa que existe una Dirección de transporte, la cual habilita de una a varias reservaciones. Todo usuario UCI tiene el derecho de realizar tanto su reservación como la de sus familiares asociados. Cada transporte posee varias reservaciones ya que los mismos están destinados a realizar viajes con cierta capacidad de personas. Cada uno de estos viajes son destinados a diferentes provincias del país y estas provincias poseen varios municipios.

Como conclusión se plantea que el modelo de dominio es aquel que se utiliza para realizar la modelación de los conceptos que participan en el negocio. En este se plasman los conceptos fundamentales para el entorno de la problemática a resolver e intentar así establecerlo como un punto de partida para el diseño de

la misma. Representa una vía para lograr la compresión del sector de negocios del sistema y posibilita identificar las interrelaciones entre las clases de dominio.

2.4. Requisitos de software

Un requisito de *software* es una característica que se debe exhibir en el *software* desarrollado o adaptado para solucionar un problema particular. A menudo los requisitos de sistemas de *software* se clasifican en funcionales y no funcionales(41).

Los **requisitos funcionales** Son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que éste debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares. En algunos casos, los requisitos funcionales de los sistemas también pueden declarar explícitamente lo que el sistema no debe hacer(42).

Los **requisitos no funcionales** son requisitos que imponen restricciones en el diseño o la implementación. Son aquellos que no hacen mención directa a las funciones específicas del sistema, sino a las propiedades emergentes que posee este como la capacidad de almacenamiento, tiempo de respuesta o la fiabilidad. Definen las restricciones del sistema como la representación de los datos, la capacidad de los dispositivos de entrada/salida, así como la disponibilidad, la protección entre otras propiedades emergentes(43).

Identificación de requisitos

Para la identificación de requisitos existen varias técnicas que permiten establecer una correcta comunicación con los interesados y su equipo de trabajo. En principio se pregunta al cliente, a los usuarios y a los que están involucrados en los objetivos del sistema o producto, luego se investiga cómo los sistemas o productos se ajustan a las necesidades del negocio, y finalmente, cómo el sistema o producto va a ser utilizado en el día a día. A continuación, es descrita la técnica utilizada para recopilar los requisitos necesarios para dar cumplimiento a la propuesta de solución(44):

- **Entrevista:** se realizó a un especialista encargado del sistema que actualmente está en funcionamiento, lo que posibilitó la interacción con el experto en el área a tratar, para arribar en conjunto a la definición de los nuevos requisitos para el sistema a desarrollar (ver anexo 1).

A partir de la utilización de las técnicas anteriores se obtuvo un total de 10 requisitos funcionales y 4 no funcionales.

Requisitos funcionales

Tabla 2. Funcionalidades del sistema.

Funcionalidad	
RF_1:	Autenticar usuario.
RF_2:	Solicitar datos de usuario autenticado.
RF_3:	Insertar solicitud de reservación.
RF_4:	Actualizar reservación.
RF_5:	Mostrar reservación.
RF_6:	Cancelar reservación.
RF_7:	Insertar solicitud de reservación de familiar.
RF_8:	Actualizar reservación de familiar.
RF_9:	Mostrar reservación de familiar.
RF_10:	Cancelar reservación de familiar.

Requisitos no funcionales

Tabla 3.Requisitos No Funcionales del sistema.

Requisitos No Funcionales	
Requerimiento de estándares	
RnF_1:	Todas las clases y funciones deben quedar bien documentadas.
Requerimiento de fiabilidad	
RnF_2:	Ante los errores que puedan ocasionarse en el sistema no se deben mostrar detalles de información que puedan comprometer su seguridad e integridad.
Requerimiento de usabilidad	
RnF_3:	La aplicación provee una interfaz visual amigable, sencilla e intuitiva, que garantiza la facilidad para el trabajo de los usuarios.
RnF_4:	Para el uso de la aplicación se requiere un dispositivo móvil con sistema operativo *Android.*

2.5. Historias de usuario

A partir de los requerimientos funcionales se han expuesto algunas de las historias de usuarios, utilizando como base las plantillas proporcionadas en la documentación de la metodología. A continuación, se detallan algunos de los campos que se definen en las mismas y se deben tener en cuenta para el posterior desarrollo de la aplicación(45):

1. En el campo NÚMERO: Se asigna un número identificador a la historia de usuario.
2. En el campo PROGRAMADOR: Se debe indicar el nombre del desarrollador encargado de implementar las funcionalidades en el sistema.
3. En el campo NOMBRE DE REQUISITO: Se asigna un nombre a la historia, para poder reconocerlo en las tareas de las siguientes fases y al momento de implementar las funcionalidades en el aplicativo.
4. El campo PRIORIDAD debe contener un valor que denote la importancia de este requerimiento para el proyecto. Este campo acepta valores referenciales recomendados de Baja, Normal y Alta.
5. El campo DESCRIPCIÓN: se describe la funcionalidad que se va a implementar en esta historia de usuario, así como los posibles escenarios de éxito y falla que puedan generarse. Esta descripción, según indican las buenas prácticas de las metodologías ágiles, debe ser lo más natural posible, sin ahondar en demasiados detalles técnicos.

Tabla 4. HU_1: Autenticar usuario.

HU	**Nombre del requisito:** Autenticar usuario	**Programador:** Eliani Brown Hernández (ejemplo)
1	**Prioridad:** Alta	**Complejidad:** Alta
	Descripción: Se validan a los usuarios mediante el servicio *LDAP* de la Universidad. Permite al usuario acceder a la aplicación con su usuario y contraseña del dominio UCI. La autenticación se realiza llenando los siguientes campos: • Nombre de usuario: Campo de texto, obligatorio. Admite todos los caracteres. Longitud entre 5 y 60 caracteres. En este campo deben poner su usuario del dominio. • Contraseña: Campo de texto, obligatorio. Admite todos los caracteres. Longitud entre 6 y 128 caracteres. En este campo deben poner su contraseña del dominio.	

	Si el usuario del dominio se autentica por primera vez, se le creará automáticamente un usuario con rol usuario autenticado, con el cual solo tendrá permisos para ver los contenidos y editar su perfil de usuario.
	Observaciones: 1. Si el usuario introduce los datos de forma incorrecta, se le emitirá un mensaje notificando el error. 2. Si el usuario deja campos obligatorios vacíos, se le emitirá un mensaje notificando el error.
	Prototipo elemental de interfaz gráfica de usuario:

Tabla 5. HU_2: Solicitar datos de usuario autenticado.

HU	**Nombre del requisito:** Solicitar datos de usuario autenticado.	**Programador:** Eliani Brown Hernández (ejemplo)
2	**Prioridad:** Alta	**Complejidad:** Alta
	Descripción: Una vez que la aplicación haya sido sincronizada, como mínimo, una vez con el Sistema de Transportación, esta acción solicita que el sistema devuelva el identificador del usuario autenticado.	
	Observaciones: No procede.	
	Prototipo elemental de interfaz gráfica de usuario:	

Tabla 6. HU_3: Insertar solicitud de reservación.

HU	**Nombre del requisito:** Insertar solicitud de reservación.	**Programador:** Eliani Brown Hernández (ejemplo)
3	**Prioridad:** Alta	**Complejidad:** Media
	Descripción: Permite a los usuarios con dominio UCI realizar la reservación de la transportación hacia sus provincias. La reservación se realiza de la siguiente manera: Se selecciona el bloque de viaje, el mismo tiene los botones reservar y cancelar. Una vez que el usuario reserve desaparecerá la opción reservar, y solo quedará la opción cancelar.	
	Observaciones: No procede.	

	Prototipo elemental de interfaz gráfica de usuario:

Tabla 7. HU_4: Actualizar reservación.

HU	**Nombre del requisito:** Actualizar reservación.	**Programador:** Eliani Brown Hernández (ejemplo)
4	**Prioridad:** Alta	**Complejidad:** Media
	Descripción: Se hace una llamada con el identificador de la reservación, esta se muestra, se modifica y luego se actualiza.	
	Observaciones: No procede.	
	Prototipo elemental de interfaz gráfica de usuario:	

Tabla 8. HU_5: Mostrar reservación.

HU	**Nombre del requisito:** Mostrar reservación.	**Programador:** Eliani Brown Hernández (ejemplo)
5	**Prioridad:** Alta	**Complejidad:** Baja
	Descripción: Permite que el usuario vea los datos de la reservación realizada, información que persistirá de manera local en la aplicación.	
	Observaciones: No procede.	
	Prototipo elemental de interfaz gráfica de usuario:	

Tabla 9.HU_6: Cancelar reservación.

HU	**Nombre del requisito:** Cancelar reservación.	**Programador:** Eliani Brown Hernández (ejemplo)
6	**Prioridad:** Alta	**Complejidad:** Media
	Descripción: Permite que el usuario cancele la reservación realizada.	

	Una vez que el usuario reserve tendrá la opción de cancelar la reservación realizada a través de un botón Cancelar que se encuentra en el bloque seleccionado. Al acceder a esta acción se muestra un mensaje de advertencia "¿Está seguro de realizar la acción? ".
	Observaciones: No procede
	Prototipo elemental de interfaz gráfica de usuario:

Tabla 10. HU_7: Insertar solicitud de reservación de familiar.

HU	**Nombre del requisito:** Insertar solicitud de reservación de familiar.	**Programador:** Eliani Brown Hernández (ejemplo)
7	**Prioridad:** Alta	**Complejidad:** Media
	Descripción: Permite al usuario con dominio UCI reservar la transportación de su familiar/es asociado.	
	Observaciones: No procede.	
	Prototipo elemental de interfaz gráfica de usuario:	

Tabla 11. HU_8: Actualizar reservación de familiar.

HU	**Nombre del requisito:** Actualizar reservación de familiar.	**Programador:** Eliani Brown Hernández (ejemplo)
8	**Prioridad:** Alta	**Complejidad:** Media
	Descripción: Se hace una llamada con el identificador de la reservación, esta se muestra, se modifica y luego se actualiza.	
	Observaciones: No procede.	
	Prototipo elemental de interfaz gráfica de usuario:	

.

Tabla 12.HU_9: Mostrar reservación de familiar.

HU	**Nombre del requisito:** Mostrar reservación de familiar.	**Programador:** Eliani Brown Hernández (ejemplo)
9	**Prioridad:** Alta	**Complejidad:** Media
	Descripción: Permite que el usuario vea los datos de la reservación de su familiar/es asociado, información que persistirá de manera local en la aplicación.	
	Observaciones: No procede	
	Prototipo elemental de interfaz gráfica de usuario:	

Tabla 13. HU_10: Cancelar reservación de familiar.

HU	**Nombre del requisito:** Cancelar reservación de familiar.	**Programador:** Eliani Brown Hernández (ejemplo)
10	**Prioridad:** Alta	**Complejidad:** Media
	Descripción: Permite que el usuario cancele la reservación realizada a su familiar/es asociado. Una vez que el usuario con dominio UCI reserve a su familiar/es, tendrá la opción de cancelar la reservación realizada a través de un botón Cancelar que se encuentra en el bloque seleccionado. Al acceder a esta acción se muestra un mensaje de advertencia "¿Está seguro de realizar la acción? ".	
	Observaciones: No procede	
	Prototipo elemental de interfaz gráfica de usuario:	

2.6. Pila de tareas o Product Backlog

Todas las historias de usuario se registran en una pila, este da lugar a la lista de Producto *(Product Backlog)*, estas historias de usuario definen a los requerimientos que se usarán para el desarrollo de la aplicación. El *Product Backl*og es una lista ordenada de todos los elementos que abarcan los requerimientos necesarios para el desarrollo de un producto y es la única fuente de requerimientos para cualquier cambio a realizarse, en estas quedan plasmadas todas las tareas generales que posteriormente se dividen entre las diferentes

iteraciones o *Sprint*(46).

Atendiendo a la importancia (valor estimado de 30 a 40) para el dueño del producto, las mismas se dividirán en tareas pequeñas o no, además se prevé un desarrollo del *software* con una estimación de 12 semanas aproximadamente. A continuación se muestra la Lista de Producto:

Tabla 14.Pila de tareas o Product Backlog.

ID	Nombre. Tarea	Importancia	Estimación (semanas)	¿Cómo probarlo?	Notas
1	Autenticar usuario.	40	2	Iniciar sección en la aplicación.	Comprobar mediante el acceso.
2	Solicitar datos de usuario autenticado.	40	1	Verificar los datos por el id del usuario devueltos.	Consultar la solicitud realizada.
3	Insertar solicitud de reservación.	40	2	Reservar en la aplicación.	Consultar reservación realizada.
4	Actualizar reservación.	40	1	Verificar la confirmación de reservación.	Consultar reservación realizada.
5	Mostrar reservación.	35	1	Solicitar los datos de la reservación realizada.	Revisar los datos mostrados.
6	Cancelar reservación.	30	1	Cancelar la reservación realizada.	Comprobar la cancelación.

7	Insertar solicitud de reservación de familiar.	40	1	Reservar en la aplicación a familiar/es asociado.	Consultar reservación realizada.
8	Actualizar reservación de reservación.	40	1	Verificar la confirmación de reservación.	Consultar reservación realizada.
9	Mostrar reservación de familiar.	35	1	Solicitar los detalles de la reservación realizada.	Revisar los datos mostrados.
10	Cancelar reservación de familiar.	30	1	Cancelar la reservación realizada.	Comprobar la cancelación.

2.6.1. Planificación de los Sprint

Sprints: consiste en unidades de trabajo que se necesitan para alcanzar un requerimiento definido en el retraso que debe ajustarse en una caja de tiempo predefinida (lo común son 30 días). Durante el *sprint* no se introducen cambios (por ejemplo, aspectos del trabajo retrasado). Así el *sprint* permite a los miembros del equipo trabajar en un ambiente de corto plazo pero estable(47).

De acuerdo a lo planteado por la metodología seleccionada se definen tres iteraciones de desarrollo, donde fueron agrupadas las HU atendiendo al nivel de importancia asignado por el cliente. Se definieron espacios de reuniones, realizadas al inicio de cada iteración, donde fueron expuestas las listas de HU a desarrollar en cada etapa.

Lista de funcionalidades por Sprint

La primera iteración comprende las HU que tributan a los requisitos de mayor importancia para el cliente, quedando organizadas en la siguiente tabla.

Tabla 15.Sprint 1 Backlog.

RF.	Nombre	Meta
1, 2, 3, 4, 7, 8	Iteración 1	Desarrollar las funcionalidades de mayor importancia y motivar al cliente.
	Descripción	
	Se seleccionaron todas las historias relacionadas con los diferentes tipos de usuarios y sus características. Estas historias son de máxima importancia para el cliente, son la solución a la problemática principal. Una vez finalizada la iteración se contará con las funcionalidades: autenticar en la aplicación y realizar la reservación.	

La segunda iteración comprende las HU correspondientes a la modificación y cancelación de las reservaciones realizadas, así como aquellas que por falta de tiempo no se desarrollen en la anterior iteración.

Tabla 16. Sprint 2 Backlog.

RF.	Nombre	Meta
6, 10	Iteración 2	Desarrollar las funcionalidades correspondientes a la cancelación de las reservaciones realizadas y las pendientes de la primera iteración.
	Descripción	
	Se seleccionaron las HU de: Cancelar reservación de la transportación. Al finalizar esta iteración el sistema deberá ser capaz de llevar a cabo el proceso de reservaciones completamente.	

La tercera y última iteración, comprende las HU propias del desarrollo de las funcionalidades correspondientes a mostrar los detalles de las reservaciones realizadas.

Tabla 17. Sprint 3 Backlog.

RF.	Nombre	Meta
9, 5	Iteración 3	Incluir funcionalidades restantes.

	Descripción
	Se seleccionaron las historias restantes, las mismas encaminadas a mostrar los detalles de las reservaciones realizadas.

2.7. Modelo de datos

Un modelo de datos es la vía para documentar un diseño de sistema de *software* complejo como un diagrama de fácil comprensión, utilizando textos y símbolos para representar la forma en que los datos necesitan fluir. El diagrama se puede utilizar como un mapa para la construcción de un nuevo *software* o para la reingeniería de una aplicación antigua(48). También está definido como un grupo de herramientas conceptuales que son utilizadas para representar determinada información en datos. Conjunto de convenciones, reglas y conceptos que posibilitan la especificación de los datos, la semántica asociada a estos, las restricciones de integridad y las relaciones existentes entre ellos(49).

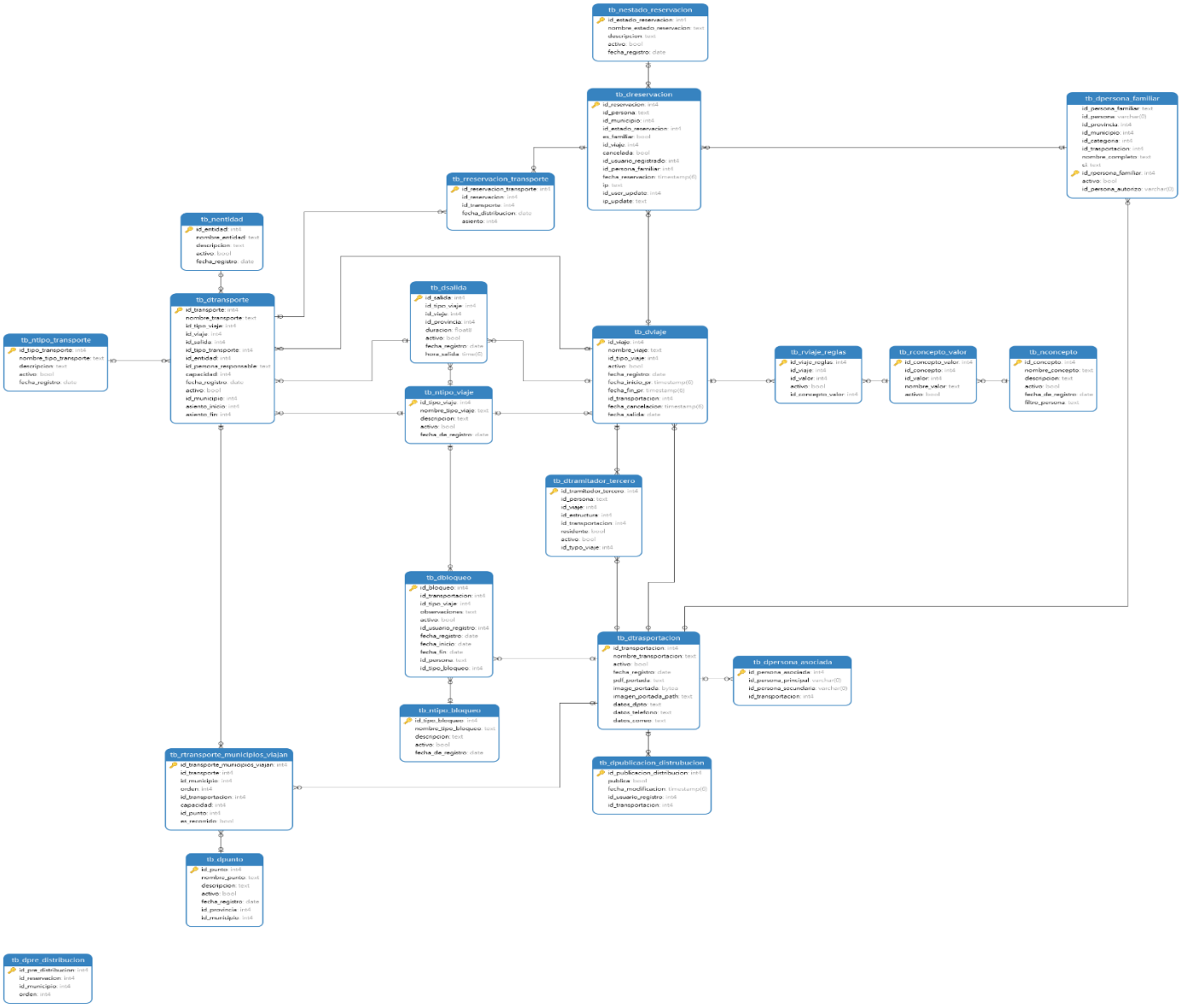

Ilustración 2. Modelo de datos del Sistema de Transportación Nacional UCI.

2.8. Arquitectura del software

Para el desarrollo de una aplicación móvil es necesario especificar la arquitectura de *software* a utilizar que se puede definir como: la estructura de un sistema o bien la forma en que va a estar organizado este; incluye

los elementos que lo conforman, sus propiedades visibles desde el exterior y las relaciones que existen entre ellos(20).

La arquitectura representa la clave para comprender, organizar y comunicar un sistema, además, facilita la evolución de la solución. Es diseñada para satisfacer los requerimientos funcionales y no funcionales establecidos por los usuarios, clientes y proveedores del sistema. Un *software* que no posee un correcto diseño arquitectónico puede funcionar de forma muy deficiente o simplemente no funcionar generando consecuencias para la organización que sirve. Para las empresas que dependen de los sistemas de información las arquitecturas de *software* son fundamentales para el logro de sus objetivos organizacionales, lo que incluye el poder evolucionar rápidamente según las condiciones altamente cambiantes de los mercados actuales(50).

Arquitectura MTV

La elección de la arquitectura se definió teniendo en cuenta el *framework* utilizado para el desarrollo de la aplicación. *Django* sigue el patrón Modelo-Vista-Controlador (MVC) tan al pie de la letra que parece ser un *framework* MVC*,* pero en la interpretación *Django* de MVC. Someramente*,* la M, V y C se separan *Django* de la siguiente manera(51):

- M es la porción de acceso a la base de datos, es manejada por la capa de la base de datos de *Django.*
- V es la porción que selecciona qué datos mostrar y cómo mostrarlos, es manejada por la vista y las plantillas.
- C es la porción que delega a la vista dependiendo de la entrada del usuario, es manejada por el *framework* mismo siguiendo *URLconf* y llamando a la función apropiada de *Python* para la *URL* obtenida.

Debido a que la "C" es manejada por el mismo *framework* y la parte más importante se produce en los modelos, las plantillas y las vistas, *Django* es conocido como un *framework MTV.* En el patrón de diseño *MTV* las siglas son definidas como:

- M significa *"Model"* (Modelo), la capa de acceso a la base de datos. Esta capa contiene toda la información sobre los datos: cómo acceder a estos, cómo validarlos, cuál es el comportamiento que tiene, y las relaciones entre los datos.

- T significa *"Template"* (Plantilla), la capa de presentación. Esta capa contiene las decisiones relacionadas a la presentación: como algunas cosas son mostradas sobre una página web u otro tipo de documento.
- V significa *"View"* (Vista), la capa de la lógica de negocios. Esta capa contiene la lógica que accede al modelo y la delega a la plantilla apropiad: se puede pensar en esto como un puente entre el modelo y la plantilla.

Se puede llegar a la conclusión de que en la interpretación de *Django* de MVC, la "vista" describe los datos que son presentados al usuario; no necesariamente el cómo se mostrarán, pero si cuáles datos son presentados(51).

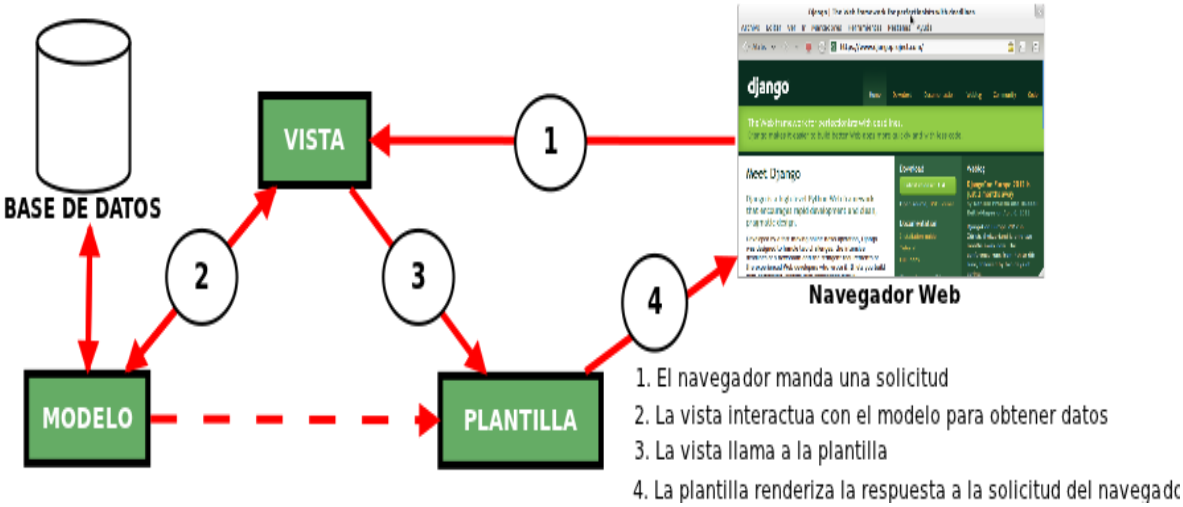

Ilustración 3. Patrón Arquitectónico MTV.

2.9. Patrones de diseño

Son valorados debido a que imponen reglas sobre la arquitectura y expresan esquemas para solucionar problemas de un mismo tipo que pueden presentarse durante el desarrollo de la aplicación. Constituyen la base para realizar la búsqueda de soluciones a problemas que se presentan en el desarrollo de software y otros marcos del diseño de interacción. Una solución es considerada un patrón de diseño cuando posee ciertas características entre las cuales se encuentran: su efectividad debe haberse comprobado resolviendo problemas similares en otras ocasiones, debe ser reutilizable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias(33).

Patrones GRASP *(Responsibility Assignment Software Patterns)*

Los Patrones Generales de Asignación de Responsabilidades de Sistemas (*GRASP*, por sus siglas en inglés) describen los principios fundamentales de la asignación de responsabilidades a objetos, de forma tal que se pueda diseñar un *software* orientado a objetos. Estos no introducen ideas novedosas, sino que son la codificación de los principios básicos más usados. Los patrones *GRASP* están compuestos por(47):

- **Bajo acoplamiento:** Se basa en proponer tener las clases lo menos ligadas entre sí, de tal forma que en caso de producirse una modificación entre alguna de ellas, se tenga la mínima repercusión posible en el resto de las clases. Potencia la reutilización del código y disminuye la dependencia.
- **Alta cohesión:** Se utiliza cuando una clase tiene una responsabilidad moderada en un área funcional y colabora con otras clases para llevar a cabo las tareas.
- **Experto:** Determina cual es la clase que debe asumir una responsabilidad a partir de la información que posee la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. De este modo se obtiene un diseño con mayor cohesión y así la información se mantiene encapsulada (bajo acoplamiento).
- **Creador:** Ayuda a identificar quien debe ser el responsable de la creación de nuevos objetos o clases.
- **Controlador:** Sirve como intermediario entre una determinada interfaz y el algoritmo que lo implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases según el método llamado. Sugiere que la lógica de negocios debe estar separada de la capa de presentación, esto permite aumentar la reutilización del código y a la vez tener un mayor control.

Patrones de Diseño GoF *(Gang of Four)*

Los patrones "Banda de los Cuatro" *(Gang-of-Four)* describen las formas comunes en que diferentes tipos de objetos pueden ser organizados para trabajar unos con otros. Tratan la relación entre clases y la formación de estructuras de mayor complejidad. Además, permiten crear grupos de objetos para ayudar a realizar tareas complejas. Existen tres tipos de patrones: de creación, estructurales y de comportamiento. Los patrones de creación abstraen la forma en la que se crean los objetos, permitiendo tratar las clases a crear de forma genérica dejando para más tarde la decisión de qué clases crear o cómo crearlas(47).

- **Instancia única *(Singleton)*:** es un patrón diseñado para restringir la creación de objetos pertenecientes a una clase u objeto. Su intención consiste en garantizar que una clase sólo tenga

una instancia y proporcionar un punto de acceso global a ella. Este patrón se refleja en las clases controladoras que son instancias únicas para la interacción entre componentes.

- **Mediador *(Mediator)*:** define un objeto que coordine la comunicación entre objetos de distintas clases. Se refleja en las librerías que funcionan como mediadoras entre las clases controladoras y los modelos de acceso a datos.
- **Observador *(Observer)*:** este patrón se encarga de definir dependencias entre objetos, de forma que, si alguno cambia su estado, automáticamente se notifica y actualizan todos los objetos que dependen de él.

2.10. Diagrama de despliegue

El diagrama de despliegue modela relaciones físicas de la arquitectura en tiempo de ejecución. No es más que en un diagrama estructurado donde se modela el *hardware* que se utiliza para implementar la plataforma web y las relaciones entre los componentes que lo conforman. Define a los artefactos como representaciones de elementos concretos en el mundo físico que son el resultado de un proceso de desarrollo(52). A continuación se muestra la figura que representa el diagrama de despliegue para la solución propuesta.

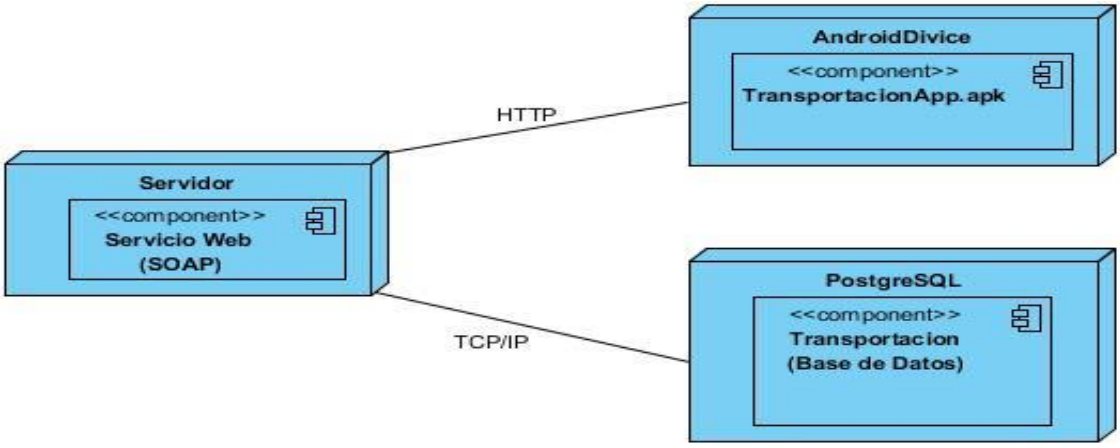

Ilustración 4.Diagrama de despliegue del sistema.

HTTP: protocolo de transferencia de hipertexto, es un estándar de red que sigue el esquema petición respuesta entre un cliente y un servidor.

TCP/IP: familia de protocolos de *Internet.* Es un conjunto de estándares de red en la que se basa la red global y que permite la transmisión de datos entre redes de computadoras sin importar el tamaño.

2.11. Conclusiones parciales

Con el análisis del proceso de Gestión de la Transportación Nacional UCI se logró realizar una identificación de los requerimientos. Se agruparon los requisitos funcionales en HU con el objetivo de definir las funcionalidades a desarrollarse en cada iteración, por lo que la planificación realizada permitirá desarrollar las mismas en un tiempo requerido. La arquitectura y demás artefactos generados permitieron diseñar una propuesta de solución, la cual utiliza varios patrones de diseño *GRASP y GOF* mediante los cuales, a partir de la implementación, se conseguirá una adecuada reutilización del código para su uso en posteriores versiones. El diseño de la BD permitirá la legibilidad, rendimiento y organización de los datos almacenados, incrementando la fiabilidad del *software* a desarrollar. Con el modelo de despliegue se pudo modelar una vista de las topologías del *hardware* sobre las que se ejecutará la aplicación.

Capítulo 3: Planificación de pruebas de la aplicación móvil Android para la reservación de la Transportación Nacional en la Universidad de las Ciencias Informáticas

En el presente capítulo se proponen las pruebas a utilizar durante la validación del correcto funcionamiento de la futura aplicación a desarrollar, con el propósito de detectar y corregir las posibles no conformidades en los usuarios.

3.1. Pruebas de software

Existen muchos tipos de pruebas, estas se realizan con el objetivo de evaluar una aplicación móvil próxima al lanzamiento en el mercado, para evitar los riesgos y la probabilidad de fallas o vulnerabilidad en un asunto determinado que se esté evaluando y que la aplicación pueda ser lanzada sin mayores complicaciones. El objetivo general de las pruebas es asegurar que se tenga un *software* de calidad, es decir, un *software* que pueda ser modificado, confiable, eficaz y pueda ser usado fácilmente. Entre estos tipos de pruebas están las que evalúan los requisitos funcionales y las que lo hacen en términos de los no funcionales(53).

Pruebas unitarias

Las pruebas unitarias sirven, como ya hemos comentado, para probar el correcto funcionamiento de una parte del código. Estas pruebas tienen como características más destacadas, que han de ser automatizables (no es obligatorio, pero sí muy recomendable), completas, reutilizables o repetibles a lo largo del tiempo, independientes entre sí y tan profesionales como el propio código. Si la prueba se centra en una parte de la aplicación que depende del dispositivo en qué se ejecuta, se la denomina prueba de integración(54).

Las pruebas unitarias agilizan el desarrollo porqué se centran en una parte del desarrollo y, por tanto, no será necesario probar dicha unidad dentro del emulador o emuladores, siempre que se trate de una parte realmente unitaria. Además, refuerzan la fiabilidad del desarrollo, ya que se realiza al mismo tiempo que la prueba que lo verifica. Y las ventajas se multiplican cuando la prueba es automatizada, pues evita la aparición de errores en un futuro al probar la aplicación de manera más exhaustiva(54).

Prueba de integración

Son una técnica sistemática para construir la arquitectura del *software* mientras se llevan a cabo pruebas para descubrir errores asociados con la interfaz. El objetivo de las pruebas de integración, es una vez

verificados los módulos unitarios por separado, ampliar el rango de alcance de las pruebas, de forma que se compruebe el comportamiento de dichos módulos, a la vez que se van integrando en el proyecto en cada iteración. Cada vez que se carga una escena nueva en la aplicación, se considera una prueba de integración, ya que depende de que se pulse un botón en la escena anterior(55).

Prueba de funcionalidad

La prueba de funcionalidad consiste en la ejecución y revisión de cada una de las funcionalidades previamente diseñadas para el aplicativo móvil, de manera que se dé a conocer el desempeño del *software* en distintos dispositivos(56).

Prueba de rendimiento

Son imprescindibles para que el producto final cause el mayor nivel de satisfacción en los usuarios. Es un tipo de prueba a la cual se somete el *software* o aplicación desarrollada, que ayuda a garantizar la compatibilidad de esta con sistemas operativos, usuarios, plataformas de *hardware* y navegadores web. Ayudan a determinar el desenvolvimiento de un sistema en determinado ambiente, el cual puede incluir *hardware,* red, sistema operativo u otro *software*(47).

Prueba de Compatibilidad

Las pruebas de compatibilidad son las encargadas de comprobar el funcionamiento del sistema en varios escenarios de ejecución. Se ejecutan en aplicaciones y *software,* para comprobar que funcionan correctamente en versiones distintas de sistemas operativos, navegadores o dispositivos con distintas características de *hardware.* Se realizan principalmente porque el *software* o aplicación puede presentar errores en dependencia del entorno donde se desarrolle(39).

Prueba de aceptación

Las pruebas de aceptación son creadas en base a las historias de usuarios, en cada ciclo de la iteración del desarrollo. El cliente debe especificar uno o diversos escenarios para comprobar que una historia de usuario ha sido correctamente implementada. Las pruebas de aceptación son consideradas como “pruebas de caja negra” *(“Black box system tests”)*. Los clientes son responsables de verificar que los resultados de estas pruebas sean correctos. Asimismo, en caso de que fallen varias pruebas, deben indicar el orden de prioridad

de resolución. Una historia de usuario no se puede considerar terminada hasta tanto pase correctamente todas las pruebas de aceptación. Dado que la responsabilidad es grupal, es recomendable publicar los resultados de las pruebas de aceptación, de manera que todo el equipo esté al tanto de esta información. Las pruebas de aceptación se conciben como la última parte del proceso de verificación, ya que se refieren a la validación del funcionamiento del sistema según los propios usuarios(55).

3.2. Conclusiones parciales

La descripción de las pruebas a utilizar para asegurar la calidad del *software*, permitirá obtener resultados satisfactorios en el posterior desarrollo de la aplicación, asegurando que la misma no contenga errores y posea la aceptación requerida.

CONCLUSIONES

Una vez culminada la investigación acerca del diseño del proceso de desarrollo de una aplicación móvil para la reservación de la transportación nacional UCI, se puede afirmar que se le dio cumplimiento al objetivo planteado, arribando así a las siguientes conclusiones:

- ✓ El análisis de los principios teóricos de la investigación, enfocado a las aplicaciones móviles para la reserva de transporte, posibilitará establecer las bases para el posterior desarrollo de la aplicación móvil para la reservación de la transportación nacional UCI, permitiendo escoger una arquitectura que proporcionará una adecuada estructura y facilitando la elección de las tecnologías y herramientas necesarias para dicho proceso.
- ✓ Las encuestas realizadas, permitieron identificar que existe un alto número de usuarios que poseen dispositivos móviles con sistema operativo *Android* en la Universidad de las Ciencias Informáticas, lo que sirvió como criterio de selección del sistema operativo más conveniente para el cual se debe desarrollar la propuesta de solución.
- ✓ A partir de la metodología seleccionada para guiar el desarrollo de la aplicación, se posibilitó la especificación de las funcionalidades del *software* por iteraciones, con el objetivo de lograr una planificación óptima de tiempo, proporcionando un mejor resultado y la obtención de una aplicación que cumpla con los requerimientos definidos.
- ✓ Las pruebas de *software* propuestas, permitirán identificar errores de implementación en la aplicación a desarrollar, pudiéndose así entregar al cliente un producto funcional y dándole cumplimiento al objetivo de esta investigación.

RECOMENDACIONES

Para garantizar el perfeccionamiento progresivo de la solución propuesta, se proponen las siguientes recomendaciones que tributarán a un producto de mejor calidad. Estas se exponen a continuación:

- Implementación de la aplicación móvil *Android* para la reservación de la transportación nacional UCI.
- Validación del correcto funcionamiento de la aplicación móvil *Android* para la reservación de la transportación nacional UCI.
- Notificar al usuario a través de la *apk* un recordatorio para realizar su reserva antes de las fechas límites establecidas por la Dirección de Transportación de la Universidad.
- Desarrollar una nueva versión de la aplicación móvil *Android* para la reservación de la transportación nacional UCI, donde se permita su acceso a través de la conexión por datos móviles.

ANEXOS

Anexo 1: Entrevista realizada al subdirector de la Dirección de Informatización.

Fecha: 12/12/2019

Hora: 2:30 p.m.

Lugar: Dirección de Informatización

Entrevistado: Ing. Jorge Arias Sojo

Preguntas de la entrevista:

1. ¿Cómo funciona el actual proceso de reservación de la transportación nacional UCI?
2. ¿Cuáles son las funcionalidades con que debe contar la aplicación móvil *Android* para la reservación de la transportación nacional UCI?

Anexo 2: Encuesta aplicada a un grupo de trabajadores y estudiantes de la UCI, más específicamente de la Facultad 1.

1. ¿Cuenta usted con un dispositivo móvil?

__Si __No

2. ¿Qué tipo de sistema operativo tiene su dispositivo móvil?

__Android __IOS __Windows Phone __Otros

3. ¿Si usted contara con una herramienta que le permitiera reservar la transportación nacional UCI desde el móvil, que se sincronice con el Sistema de Gestión de Transportación Nacional UCI, con la ventaja de que lleve guardada consigo la última actualización de la reservación realizada, por donde realizaría usted la reserva?

__dispositivo móvil __PC

4. ¿Cree usted que sería útil que se desarrollara una aplicación móvil Android, para realizar la reservación de la transportación nacional UCI desde un dispositivo móvil?

__Si __No

Anexo 3: Total de estudiantes y trabajadores encuestados.

Tabla 18.Total de estudiantes y trabajadores encuestados.

Áreas	Cantidad de trabajadores y estudiantes a encuestar	Cantidad de trabajadores y estudiantes ausentes	Total

Centro de Innovación y Desarrollo de Internet (CIDI)	42 trabajadores	8 trabajadores	50
Grupos de 1er año de la Facultad 1	133 estudiantes	7 estudiantes	140
Grupos de 4to año de la Facultad 1	102 estudiantes	10 estudiantes	112
Grupos de 5to año de la Facultad 1	94 estudiantes	5 estudiantes	99
Total	371	30	401

<u>Anexo 4:</u> Cantidad de estudiantes y trabajadores que creen es necesario desarrollar una aplicación para realizar la reserva de transporte desde su móvil.

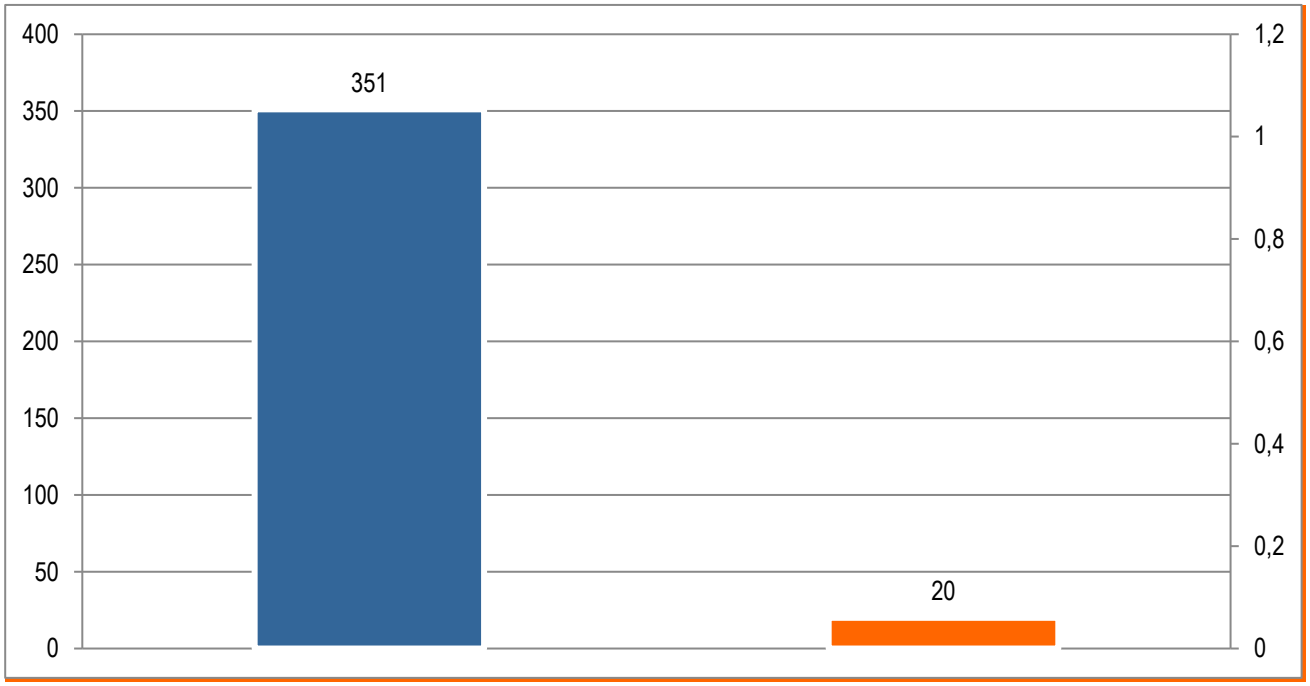

Ilustración 5.Cantidad de estudiantes y trabajadores que creen necesario desarrollar una aplicación para realizar la reserva de transporte desde su móvil.

<u>Anexo 5:</u> Cantidad de usuarios con distintos sistemas operativos.

Tabla 19.Cantidad de usuarios con distintos sistemas operativos.

Áreas \ **Sistema Operativo**	**Android**	**IOS**	**Windows Phone**	**Otros**
Centro de Innovación y Desarrollo de Internet (CIDI)	29	4	0	9
Grupos de 1er año de la Facultad 1	125	6	2	0
Grupos de 4to año de la Facultad 1	94	1	3	4
Grupos de 5to año de la Facultad 1	87	5	0	2
Total	335	16	5	15

REFERENCIAS BIBLIOGRÁFICAS

1. KARELL, Denis Sixto Francia and VALDÉS, Osviel Rodríguez. Apk para restringir el acceso a aplicaciones por usuarios no autorizados en sistema operativo Andoid. .

2. QUISHPE COLLAGUAZO, Edgar Hipólito. *Aplicacón móvil para consulta de turnos de regadío del Canal de Riego Píllaro Ramal Norte Yatchil Central.* Ecuador : Universidad Técnica de Ambato., 2017.

3. Definicion .De. [online]. Available from: https://definicion.de/reserva/

4. CUBA PIÑEIRO, Dairon Sabad and BRUZÓN PÉREZ, José Gabriel. *Aplicación móvil Android para realizar la reserva de alimentos de los trabajadores de la Universidad de las Ciencias Informáticas .* La Habana. : Universidad de las Ciencias Informáticas., 2016.

5. Universidad de la Ciencias Informáticas. *UCI* [online]. Available from: www.uci.cu

6. FERNÁNDEZ RABILERO, Ramón and VARONA CARMENATES, Arian. *Sistema para la Gestión de las Transportaciones Nacionales de la Universidad de las Ciencias Informáticas.* La Habana. : Universidad de las Ciencias Informáticas., 2016.

7. Concepto de Aplicación. *Concepto.de.* [online]. 2015. Available from: http://concepto.de/aplicacion/.

8. DELÍA, Lisandro Nahuel. *Desarrollo de aplicaciones móviles multiplataforma*. PhD Thesis. Facultad de Informática, 2017.

9. LUNA, Fernando, MILLAHUAL, Claudio Peña and IACONO, Matías. *PROGRAMACION WEB Full Stack 20 - Expandir Mobile web*. RedUsers, 2018. Google-Books-ID: 1JdGDwAAQBAJ

10. SERNA, Sebastián. *Diseño de interfaces en aplicaciones móviles*. Grupo Editorial RA-MA, [no date].

11. MANTILLA, Maira Cecilia Gasca, ARIZA, Luis Leonardo Camargo and DELGADO, Byron Medina. Metodología para el desarrollo de aplicaciones móviles. *Tecnura*. 2014. Vol. 18, no. 40, p. 20–35.

12. *Linux*. USERSHOP, [no date]. ISBN 978-987-1773-12-1. Google-Books-ID: vDek7X3Jwm8C

13. DAVID, Robledo. *Desarrollo de aplicaciones para Android I*. Ministerio de Educación, Cultura y Deporte, 2016. ISBN 978-84-369-5687-0. Google-Books-ID: PHmbDQAAQBAJ

14. Desarrollo en Kotlin para la creación de una red social. [online]. [Accessed 28 April 2020]. Available from: https://ebuah.uah.es/dspace/handle/10017/33859

15. Uber - Aplicaciones en Google Play. [online]. [Accessed 20 March 2020]. Available from: https://play.google.com/store/apps/details?id=com.ubercab&hl=esLibrary Catalog: play.google.com

16. inDriver - Viajes rentables. Taxi alternativo - Aplicaciones en Google Play. [online]. [Accessed 20 March 2020]. Available from: https://play.google.com/store/apps/details?id=sinet.startup.inDriver&hl=esLibrary Catalog: play.google.com

17. YEGO Mobility - Aplicaciones en Google Play. [online]. [Accessed 20 March 2020]. Available from: https://play.google.com/store/apps/details?id=com.getyugo.app&hl=esLibrary Catalog: play.google.com

18. Aplicaciones para Android de Bolt Technology en Google Play. [online]. [Accessed 20 March 2020]. Available from: https://play.google.com/store/apps/dev?id=8420210619522248974&hl=esLibrary Catalog: play.google.com

19. [Compartiendo Apk] Viajeros, una apk para llegar a cualquier parte. |. [online]. 10 January 2017. [Accessed 28 April 2020]. Available from: https://jorgen.cubava.cu/2017/01/10/compartiendo-apk-viajeros-una-apk-para-llegar-a-cualquier-parte/Library Catalog: jorgen.cubava.cu

20. Lanzan nueva aplicación para conocer la disponibilidad de pasajes de la empresa Viajero (+ Descarga). *Cubadebate* [online]. 2 April 2019. [Accessed 28 April 2020]. Available from: http://www.cubadebate.cu/noticias/2019/04/02/lanzan-nueva-aplicacion-para-conocer-la-disponibilidad-de-pasajes-de-la-empresa-viajero-descarga/Library Catalog: www.cubadebate.cu

21. Disponible aplicación móvil sobre el servicio de trenes nacionales (+ Apk). *Cubadebate* [online]. 13 July 2019. [Accessed 28 April 2020]. Available from: http://www.cubadebate.cu/noticias/2019/07/13/disponible-aplicacion-movil-sobre-el-servicio-de-trenes-nacionales-apk/Library Catalog: www.cubadebate.cu

22. PRESSMAN, Roger S. *Ingeniería de Software, un enfoque práctico.* 2002. ISBN 84-481-3214-9.

23. ORDÓÑEZ, Mariuxi Paola Zea, RÍOS, Jimmy Rolando Molina and CASTILLO, Fausto Fabían Redrován. *ADMINISTRACIÓN DE BASES DE DATOS CON POSTGRESQL*. 3Ciencias, 2017. ISBN 978-84-946684-6-3.

24. YÉLAMOS SAN ANDRÉS, Álvaro. Aplicación Android para buses urbanos de Guadalajara. [online]. 2018. [Accessed 30 April 2020]. Available from: https://ebuah.uah.es/dspace/handle/10017/33948Accepted: 2018-07-27T10:17:13Z

25. DIAZ, Guevara and OMAR, Darwin. Sistema de gestión de inventario basado en la teoría de inventarios y control de producción utilizando tecnología QR, para mejorar la gestión del inventario en la empresa Ecovive SAC. *Universidad Católica Santo Toribio de Mogrovejo - USAT* [online]. 2019. [Accessed 30 April 2020]. Available from: http://tesis.usat.edu.pe/handle/20.500.12423/2363Accepted: 2020-02-11T13:52:48Z

26. MONTERO, Moreno and ALEJANDRO, Mario. Aplicación web para la generación de solicitudes académicas de la Pontificia Universidad Católica del Ecuador Sede Ambato. [online]. 2017. [Accessed 28 April 2020]. Available from: https://repositorio.pucesa.edu.ec/handle/123456789/1929Accepted: 2017-05-09T20:08:30Z

27. CUESTA PLÚA, María Belén. *Desarrollo de un prototipo de sistema de control en la nube para gestionar el préstamo de bicicletas empleando Android*. Quito, 2020., 2020.

28. FENOLLAR ONRUBIA, Daniel. Aplicación móvil para la gestión de pacientes con Mieloma Múltiple. [online]. 14 October 2019. [Accessed 28 April 2020]. Available from: https://riunet.upv.es/handle/10251/128258Accepted: 2019-10-14T17:32:33Z

29. Servicio multiplataforma para la consulta y gestión de contenido multimedia. [online]. [Accessed 28 April 2020]. Available from: https://riuma.uma.es/xmlui/handle/10630/13368

30. PALACIOS RAMÍREZ, José Antonio. Servicio multiplataforma para la consulta y gestión de contenido multimedia. . 2017.

31. POSINCOVICH, ALAN MARK SOUSA. PROYECTO FIN DE GRADO. .

32. SANTOS, Crescencio Bravo and DUQUE, Miguel Ángel Redondo. *Sistemas interactivos y colaborativos en la web*. Univ de Castilla La Mancha, 2004. ISBN 978-84-8427-352-3. Google-Books-ID: 2V9WB5s9lU4C

33. *Manual de desarrollo web con Grails*. Imaginaworks Software Facto, 2009. ISBN 978-84-613-2651-8. Google-Books-ID: Wa9AXT7zHVwC

34. Aplicación de servicios web SOAP/REST para funcionalidades existentes en sistemas informáticos provinciales. [online]. [Accessed 25 May 2020]. Available from: http://sedici.unlp.edu.ar/handle/10915/94248

35. GARRIDO MOSCOSO, Jesús. Servicio multiplataforma para la consulta y gestión de contenido multimedia. [online]. 23 March 2017. [Accessed 28 April 2020]. Available from: https://riuma.uma.es/xmlui/handle/10630/13368Accepted: 2017-03-23T12:35:15Z

36. MOLL, Mariano. Aplicación para facturación de energía eléctrica. [online]. 2019. [Accessed 28 April 2020]. Available from: https://repositorio.uesiglo21.edu.ar/handle/ues21/16730Accepted: 2019-10-30T22:12:09Z

37. SOMMERVILLE, Ian. *Ingeniería de software*. 2005.

38. CADAVID, Andrés Navarro, MARTÍNEZ, Juan Daniel Fernández and VÉLEZ, Jonathan Morales. Revisión de metodologías ágiles para el desarrollo de software. *Prospectiva*. 2013. Vol. 11, no. 2, p. 30–39.

39. PRESSMAN, Roger S. *Ingeniería del software, un enfoque práctico.* New York. EEUU, 2011. ISBN 978-0-07-126782-3.

40. GARCÍA-PEÑALVO, F. J. and GARCÍA-HOLGADO, A. Análisis orientado a objetos. *Recursos docentes de la asignatura Ingeniería de Software I. Grado en Ingeniería Informática. Curso*. 2017. Vol. 2018.

41. SOMMERVILLE, Ian. *Ingeniería de Software*. Séptima edición. 2008. ISBN 978-0-321-31379-9.

42. TORO LAZO, Alonso and GÁLVEZ BOTERO, Juan Guillermo. Procedimiento para especificar y validar requisitos de software en mipymes desarrolladoras de software de la ciudad de Pereira, basado en estudios previos en la región. [online]. 2017. [Accessed 24 May 2020]. Available from: http://repositorio.autonoma.edu.co/xmlui/handle/11182/103Accepted: 2018-11-09T15:07:14Z

43. ALAMO, Junior Calle. INSTITUTO TECNOLOGICO SUPERIOR ESCÁRCEGA Organismo Público Descentralizado de la Administración Pública del Estado de Campeche “Reporte de instalación de apache” Materia. [online]. [Accessed 24 May 2020]. Available from: https://www.academia.edu/4560438/INSTITUTO_TECNOLOGICO_SUPERIOR_ESC%C3%81RCEGA_Organismo_P%C3%BAblico_Descentralizado_de_la_Administraci%C3%B3n_P%C3%BAblica_del_Estado_de_Campeche_Reporte_de_instalaci%C3%B3n_de_apache_Materia_Planificaci%C3%B3n_y_modelado_Alumno_Olivera_Sosa_%C3%81ngel_Gabriel_Profesor

44. PRESSMAN, Roger S. *Ingeniería de software: Un enfoque práctico.* VI. Madrid, 2005.

45. CRUZADO, Javier Gamboa, UCHAMACO, Guido Raúl Larico, SOTO, Luis Soto, MALASQUEZ, Naysha Chacón, ACHULLE, José Tuiro and CHAMBI, Sandro Guzman Canahuire. Aplicación móvil de realidad aumentada, utilizando la metodología MOBILE–D, para el entrenamiento de técnicos de mantenimiento de maquinaria pesada en la empresa ZAMINE SERVICE PERU SAC. *Ceprosimad*. 2017. Vol. 5, no. 2, p. 39–51.

46. CEVALLOS, Ramírez and RUPERTO, Willington. Estudio del Framework Meteor para aplicaciones web. desarrollo del sistema de planificación de proyectos para el GAD parroquial La Esperanza. [online]. 10 October 2018. [Accessed 25 May 2020]. Available from: http://repositorio.utn.edu.ec/handle/123456789/8612Accepted: 2018-10-10T16:26:32Z

47. PRESSMAN, Roger S. *Ingeniería del software, un enfoque práctico.* VII. México, 2010. ISBN ISBN: 978-607-15-0314-5.

48. ROUSE, M. ¿Qué es Modelado de datos? *Definición en WhatIs.com. TechTarget* [online]. Available from: https://searchdatacenter.techtarget.com/es/definicion/Modelado-de-datos.

49. AGUIRRE AYO, Fabián Patricio. *Sistema de levantamiento de activos fijos en Android para la empresa Cayman Systems.* Quito, 2018.

50. Importancia de la arquitectura de software en las organizaciones. [online]. [Accessed 24 May 2020]. Available from: https://www.icesi.edu.co/unicesi/todas-las-noticias/1949-importancia-de-la-arquitectura-de-software-en-las-organizacionesLibrary Catalog: www.icesi.edu.co

51. HOLOVATY, Adrian and KAPLAN-MOSS, Jacob. *El libro de Django 1.0.* [online]. [no date]. [Accessed 7 March 2020]. Available from: https://uniwebsidad.com

52. HERNÁNDEZ, Domingo and BAHENA, Carlos. Arquitectura para el Despliegue Tridimensional en Dispositivos Móviles de Datos Generados por Tomógrafos. *Tekhné*. 2017. Vol. 1, no. 19.

53. GAVIRIA PULGARÍN, Hermes Duvier and CARMONA, Jhon Fernando. Metodología de testing de seguridad para aplicaciones móviles android, en el campo de la salud. . 2018.

54. VIQUE, Robert Ramírez. Métodos para el desarrollo de aplicaciones móviles. *PID_00176755*. 2012.

55. COBO FERNÁNDEZ, Guillermo. Desarrollo de una aplicación móvil de realidad virtual para el aprendizaje en las aulas. . 2017.

56. CHÁVEZ, Hugo and CHAVEZ, Jesús. Aplicación Móvil en Realidad Aumentada de Apoyo para el Aprendizaje de Verbos Irregulares en inglés. *REVISTA CIENTÍFICA UNE*. 2019. Vol. 3, no. 3, p. 34–39.