UNIVERSIDAD ANDINA DEL CUSCO

FACULTAD DE INGENIERÍA Y ARQUITECTURA

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

TESIS:

DESARROLLO DE UNA APLICACIÓN MÓVIL PARA CONSULTAS DE RUTAS DEL TRANSPORTE PÚBLICO EN LA CIUDAD DEL CUSCO

Presentado por:
BR. DAVID JOSUE MACHICAO VALENCIA

Para optar por el Título Profesional de
INGENIERO DE SISTEMAS

Asesora:
MG. MARIA ISABEL ACURIO GUTIERREZ

CUSCO – PERÚ

2018

PRESENTACIÓN

Señor Decano de la Facultad de Ingeniería y Arquitectura de la Universidad Andina del Cusco.

Señores Docentes miembros del Jurado.

De acuerdo a lo dispuesto por el reglamento de Grados y Títulos vigente de la Escuela Profesional de Ingeniería de Sistemas de la Facultad de Ingeniería y Arquitectura de la Universidad Andina del Cusco, para optar el título profesional de Ingeniero de Sistemas pongo a vuestra consideración el presente trabajo de investigación que lleva por título:

"DESARROLLO DE UNA APLICACIÓN MÓVIL PARA CONSULTAS DE RUTAS DEL TRANSPORTE PÚBLICO EN LA CIUDAD DEL CUSCO"

DEDICATORIA

A mis padres, por todo y por siempre.

A mis hermanos y a mi sobrina.

A mis abuelos.

A mi familia.

A mis amigos.

AGRADECIMIENTOS

Agradezco por sobre todas las cosas a mis padres, David Machicao Olivera y Mirian Luz Valencia Acosta, por su esfuerzo, paciencia y amor incondicional. También a mis hermanos, (Vanessa, Sharon y Enmanuel) y a mi sobrina (Micaela), por brindarme inspiración para cumplir las metas.

A mis abuelos y mi tío Ronald por el apoyo y el cariño, además de la confianza depositada en mi persona, y por supuesto, agradecer a mi familia entera.

A mi asesora, por el tiempo brindado, la paciencia, la guía y los consejos.

A la Universidad Andina del Cusco, por brindarme el conocimiento para continuar con mi camino

A mis amigos, todos ellos buenos amigos.

RESUMEN

El objetivo de la presente tesis ha sido el Desarrollo de una Aplicación Móvil para Consulta de Rutas del Transporte Público en la Ciudad del Cusco. Para lo cual se ha seguido un procedimiento ordenado y cumplido con los requerimientos encontrados.

Se utilizó como metodología de desarrollo de software al Proceso Unificado de Desarrollo de Software, y se acompañó al mismo con el marco de trabajo Scrum para agilizar el proceso de desarrollo y documentación, además de aprovechar las ventajas que el uso de las metodologías ágiles trae al proceso de desarrollo de software.

Se investigó lo necesario para desarrollar un aplicativo para Android que sea utilizado por los clientes; y un sistema Web para dar mantenimiento a las tablas creadas para ser usado por el desarrollador. Dentro de la investigación se incluyó la creación y consumo de Servicios Web, creación y manipulación de bases de datos SQLite, solicitud de permisos en Android, manejo de mapas de Google tanto para Web como para Android, entre otros temas.

El aplicativo móvil es capaz de obtener las coordenadas de origen y destino del viaje del usuario, sugerir la(s) empresa(s) a utilizar o sugerir llegar al destino a pie, graficar en los mapas de Google la ruta que se va a seguir mientras se viaja en bus, indicando qué paraderos debe usar el usuario para embarcar y desembarcar, y graficando las rutas que debe seguir a pie el usuario desde el punto de origen al paradero de subida; y desde el paradero de bajada hacia el punto de destino. Finalmente, el aplicativo también es capaz de indicar la ubicación en tiempo real de usuario mediante el uso de GPS en los mapas de Google.

El sistema web es capaz de dar mantenimiento a la base de datos creada mediante interfaces amigables; se mantiene los paraderos, empresas, y rutas registradas; en el caso particular de estas últimas, se mantienen las rutas registradas junto a todas

las coordenadas que forman parte de la misma tanto de ida como de vuelta, y a los paraderos que forman parte de cada ruta en ambos sentidos de manera ordenada.

La respuesta de las personas a quienes se mostró el aplicativo ha sido muy alentadora, dentro de los resultados de la encuesta realizada se destaca especialmente la sencillez con la que se puede llegar a un resultado.

ABSTRACT

The objective of the current thesis has been the Development of a Mobile App for Route's Lookup of the Public Transportation in Cusco City. For which, it has been followed an orderly procedure and acoomplished all the found requirements.

It was used, as software development methodology, The Unified Software Development Process, and it was accompanied by the framework Scrum as to speed up the development process and documentation, in addition to leveraging the advantages that agile methodologies bring to software development process.

It was researched all the necessary in order to develope an Android application for the use of the users, and a Web system to maintain the created tables for the use of the software developer. It was included inside the research; the creation and consumption of Web Services, creation and handling of SQLite's databases, Android's request for permissions, Google maps handling both for Web and Android, among other topics.

The mobile app is capable of getting the user's origin and destination coordinates, suggest the transport company(ies) to use or suggest walk, graph on Google maps the route to follow while travelling in the bus, pointing out which bus stops the user have to use to board and disembark, and graphing the route the user must follow from the origin point to the boarding bus stop and from the disembarking bus stop to the destination point. Finally, the app is capable of indicating the users current real location by using GPS on Google's maps.

The Web system is capable of providing maitenance to the created database through friendly interfaces, bus stops are maintained, companies, and registered routes, in the particular case of the last ones, routes are maintained with all the coordinates that are part of it both round trip, and the bus stops that are part of the routes in both ways; orderly.

The response of the people whom the app was shown has been very encouraging, within the results of the survey applied, the simplicity to get to a result stands out particulary.

ÍNDICE GENERAL

Índice de Tablas

Índice de Figuras

Índice de Anexos

INTRODUCCIÓN

El presente trabajo tiene como objetivo el Desarrollo de una Aplicación Móvil para Consultas de Rutas del Transporte Público en la Ciudad del Cusco, y ha sido elaborado para optar por el título profesional de Ingeniero de Sistemas, extendido por la Escuela Profesional de Ingeniería de Sistemas de la Universidad Andina del Cusco.

El transporte público y la penetración de dispositivos móviles han crecido en la provincia del Cusco. Hoy en día se puede ver que los buses han reemplazado a las llamadas “combis”; y que el sistema de transporte público va creciendo debido a la demanda que este tiene. Lo propio ocurre en la penetración de teléfonos inteligentes dentro de los clientes habituales, y también potenciales, de las distintas empresas de transporte. El número de usuarios de teléfonos inteligentes va en aumento, y los planes de las distintas operadoras móviles brindan cada vez más y más gigabytes para poder hacer uso del internet.

Para poder viajar en alguna de las empresas de transporte público dentro de la provincia del Cusco es necesario tener conocimientos previos. Un usuario nuevo difícilmente hará uso del mismo debido a la falta de información que existe, incluso, un usuario habitual; deberá preguntar a otros usuarios para dirigirse a alguna zona de la ciudad que no frecuenta; ya que no conoce qué empresas pasan por allí.

La presente tesis busca aliviar la falta de información existente en la provincia del Cusco, en cuanto al sistema de transporte público, mediante el desarrollo de una aplicación para Android que permita a sus usuarios básicamente dos cosas. La primera es; consultar sobre qué empresas y rutas existen en la ciudad, y la segunda; recibir información sobre qué empresas puede utilizar para llegar a su destino, el tiempo estimado de viaje, qué paraderos debe utilizar tanto para abordar como para desembarcar y por qué calles se va movilizar. Todo ello mediante interfaces gráficas sencillas de utilizar, y haciendo uso de los mapas de Google y el GPS de los teléfonos inteligentes.

El primer capítulo trata sobre aspectos generales, ámbito de influencia, objetivos, descripción del problema, hipótesis, justificación, metodología y matriz de consistencia. El segundo lista el marco teórico utilizado, donde se encuentra la explicación de los diferentes elementos y términos usados. También explica los antecedentes de la investigación. El tercer capítulo explica la metodología utilizada, incluyendo datos sobre la población de muestra, instrumentos y recolección de datos.

El cuarto capítulo ingresa directamente en el desarrollo de la tesis, lista los objetivos que se busca obtener a medida que se va desarrollando tanto el aplicativo móvil como un sistema web para dar mantenimiento al sistema. El quinto capítulo hace un listado de los resultados obtenidos. El sexto capítulo se discute los resultados obtenidos en comparación con los obtenidos en las tesis de referencia, además de los antecedentes que se mencionan en la presente tesis.

CAPÍTULO 1 – ASPECTOS GENERALES

1.1 Ámbito de la Influencia de la Tesis

La presente tesis tiene como ámbito de Influencia los 8 distritos de la provincia del Cusco, éstos son: Cusco, Wanchaq, Santiago, San Sebastián, San Jerónimo, Saylla, Poroy y Ccorca, sin embargo por fines prácticos al momento de hacer las pruebas, se tomaron en cuenta sólo 3 rutas, las cuales cubren los distritos de San Jerónimo, San Sebastiá, Wanchaq, Cusco y Santiago.

1.1.1 Ámbito de la influencia teórica

De acuerdo a las líneas de investigación de la Escuela Profesional de Ingeniería de sistemas, la presente tesis tiene como línea de investigación a las Tecnologías de Información, dado que el Desarrollo de una Aplicación Móvil (de ahora en adelante app) para Consultas de Rutas del Transporte Público en la ciudad del Cusco; implica también, además de la aplicación misma, el desarrollo de un software para Web. Para ello se hace uso de los lenguajes de programación Java, Javascript y PHP, el lenguaje de marcado HTML, SQL como lenguaje de manipulación, control y definición de datos, CSS como lenguaje de estilos, dos gestores de bases de datos (MySql y SQLite), tecnología disponible brindada por Google y diferentes programas informáticos.

Vale la pena dejar en claro que se toma como metodología para el desarrollo de la presente tesis al Proceso Unificado de Desarrollo de Software (de ahora en adelante PUDS), junto al marco de trabajo SCRUM para agilizar tareas y aprovechar las potencialidades de ambos.

1.2 Descripción de la situación actual

La provincia del Cusco, de acuerdo a información encontrada en la página web del Gobierno Municipal del Cusco, cuenta con 38 empresas de transporte, cada una de ellas con su respectivo contrato de concesión de ruta (Contratos de

Concesión de Rutas). Estas empresas brindan el servicio de transporte público, uniendo los 8 distritos de la provincia del Cusco mediante las rutas existentes.

El proceso para movilizarse haciendo uso de transporte público de la provincia del Cusco es el siguiente. Acercarse al paradero útil más cercano (al paradero por donde pase la empresa que lleve a uno al destino que desee), viajar, y finalmente bajarse en el paradero dentro de la ruta de la empresa elegida que se ubique a la menor distancia del destino elegido. Todo suena simple, sin embargo hay un factor importantísimo que permite realizar este proceso y es el conocimiento previo de las rutas, empresas y ubicación de paraderos dentro de provincia del Cusco.

Sin este conocimiento previo; resulta complicado movilizarse dentro la ciudad del Cusco, ya que se debe preguntar a terceras personas, aventurarse y conocer por uno mismo el sistema. Hasta el año 2017 aproximadamente era posible encontrar las distintas rutas en la página web del Gobierno Municipal del Cusco, sin embargo ahora solamente se encuentran los contratos. La suma de todo esto, complica el uso del sistema de transporte público del Cusco a personas que salgan de sus rutas habituales, y mucho más a aquellos que vienen de visita.

1.3 Formulación del problema

¿Cómo obtener rápidamente información sobre las rutas del transporte público dentro de la ciudad del Cusco?

1.3.1 Descripción del problema

Los habitantes y visitantes de la ciudad del Cusco se movilizan dentro de la misma haciendo uso de distintos medios de transporte. Para los usuarios habituales del servicio de transporte público de la ciudad, es relativamente sencillo llegar a sus destinos haciendo uso del mismo, se mueven dentro de las rutas que conocen, saben dónde se encuentran los paraderos que necesitan y conocen los buses que debe abordar para llegar a sus destinos.

Hasta ese punto el movilizarse es sencillo, pero se empieza a complicar cuando los usuarios deben viajar a zonas alejadas de sus rutas habituales, se empieza a tener dudas sobre qué buses pueden abordar y dónde se encuentran sus paraderos, lógicamente se puede hacer uso de taxis o transporte privado, pero estos cuestan mínimamente 3.75 veces más que viajar en el transporte público (viajar en bus cuesta 0.80 soles, mientras que viajar en taxi cuesta mínimo 3 soles), además, como es de conocimiento común, viajar en taxi o un auto privado contribuye a generar mayor contaminación y tráfico en las avenidas.

Si para un usuario habitual surgen algunos problemas al salir de sus rutas habituales, para los visitantes nacionales o extranjeros que conocen poco o nada de la ciudad; es mucho más complicado, se ven obligados a usar cualquier otro medio de transporte o preguntar a otras personas y aventurarse en el sistema de transporte público; viajando con la evidente inseguridad que causa no saber por dónde se va, cómo regresar, cuánto falta para llegar y obviamente el no saber dónde se encuentran en ese momento.

1.4 Objetivos

El objetivo general y los objetivos específicos son detallados a continuación:

1.4.1 Objetivo General

Desarrollar una aplicación móvil para consultas de rutas del transporte público en la ciudad del Cusco.

1.4.2 Objetivos Específicos

El desarrollo de la aplicación móvil para consulta de rutas debe cumplir con los siguientes objetivos específicos.

Objetivo específico 1: Desarrollar un sistema backend basado en web para dar mantenimiento a las tablas de paraderos, rutas, empresas, coordenadas de ruta (de ida y vuelta) y paraderos de ruta (ida y vuelta).

Objetivo específico 2: Mostrar en el sistema backend mediante consultas a la base de datos y uso de los mapas de Google; los gráficos de las rutas ingresadas para poder probarlas, así como también los paraderos que son utilizados por las rutas.

Objetivo específico 3: Desarrolllar una aplicación para Android donde el usuario pueda indicar los puntos de origen y destino de su viaje.

Objetivo específico 4: Mostrar en la app Android la ubicación del usuario, el listado de empresas y rutas registradas, y el gráfico de las rutas de ida y vuelta de las empresas.

Objetivo específico 5: Sugerir empresas al usuario mediante la app para que este llegue a su destino, dependiendo de los puntos de origen y destino ingresados. En caso la distancia sea muy corta entre ambos puntos o no existan rutas para cubrir esos puntos, se debe sugerir ir a pie y graficar la ruta a seguir.

Objetivo específico 6: Graficar, en la app, la ruta que el usuario seguirá en el bus, además del camino desde el punto de origen al paradero de subida, y el camino entre el paradero de bajada hasta el punto de destino.

Objetivo específico 7: Realizar, dentro de la app, seguimiento en tiempo real de la ubicación del usuario y mostrárselo para que este pueda ubicarse en todo momento durante su viaje y pueda bajar en el lugar correcto.

1.5 Hipótesis

El desarrollo de una aplicación móvil para consultas de rutas del transporte público de la ciudad del Cusco, brindará a sus usuarios de manera rápida la información necesaria para saber qué empresa tomar desde y hasta qué lugar, para así poder movilizarse dentro de la ciudad haciendo uso del servicio de transporte público.

1.6 Justificación

El departamento del Cusco se divide en 13 provincias, y la provincia del Cusco, punto central de la presente investigación, se divide en 8 distritos, los cuales son: Cusco, Wanchaq, Santiago, San Sebastián, San Jerónimo, Saylla, Poroy y Ccorca. Todos ellos interconectados por las rutas de transporte público que cubren las distintas empresas. El aumento de la población genera mayor crecimiento de la ciudad, a esto podemos agregar el creciente número de visitantes que tiene el Cusco, y podemos concluir que al aumentar ambos, la necesidad de transportarse; también aumenta.

Según datos del INEI (Instituto Nacional de Estadística e Informática), en sus publicaciones (Estado de la Población Peruana 2015) nos indica que las poblaciones del Perú y del Cusco fueron de 31,151,643 y 1,316,729 habitantes respectivamente (pág 3), y que la población de la provincia del Cusco en ese mismo año fue de casi casi medio millón de habitantes, para ser mas exactos; 450,095 (pág. 7). Por otra parte, la Dirección Regional de Comercio Exterior y Turismo del Cusco (DIRCETUR) nos indica que el número total de visitantes de nuestra ciudad durante el año 2015 (Boletín Estadístico de Turismo 2015), fue de 2,881,677, de los cuales 1,023,419 fueron nacionales y 1,858,258 extranjeros (pág. 12).

De acuerdo a la página web del Gobierno Municipal del Cusco, existen 38 empresas de transporte en nuestra ciudad; cada una con su respectivo contrato (Contratos de Concesión de Rutas), estas cubren las necesidades de transporte dentro de los 8 distritos de la provincia del Cusco transportando a las personas dentro de sus rutas concesionadas. Para hacer uso de las distintas líneas, además del pago que se debe hacer, es bueno conocer por dónde van, especialmente si se es visitante, o se esta viajando a áreas poco conocidas de la ciudad.

Desarrollar una Aplicación Móvil para consultas de rutas del Transporte Público en la ciudad del Cusco, es una excelente solución al problema de la falta de información tanto para habitantes y visitantes sobre cómo hacer uso del servicio

de transporte público de Cusco, además de poder visualizar las rutas de las distintas empresas y conocer dónde se encuentran los distintos paraderos dentro de los diferentes distritos.

También se debe tener en cuenta que el desarrollo de una aplicación móvil de estas características no es una tarea aislada, ya que se debe crear, como parte del proyecto; un programa, en este caso web, que permita realizar las distintas tareas de mantenimiento a una base de datos en línea. Consecuentemente, la investigación que se debe llevar a cabo no se centra solamente en Android (que es el sistema operativo que se usará en el presente proyecto para la elaboración de la app); y en todo lo que trae consigo este sistema operativo, sino también en algunos otros lenguajes que permitan el mantenimiento de una base de datos en línea y el manejo de los mismos, en este caso PHP, Javascript y SQL.

Por todo lo expuesto, el presente proyecto no solo busca dar solución al problema que tienen los habitantes y visitantes cuando desean hacer uso de el sistema de transporte público, sino también dar mantenimiento y capacidad de pruebas inmediatas a un sistema de estas características; investigando todo lo necesario (y un poco más) para lograr el cumplimiento de los objetivos, ampliando el conocimiento en programación web y Android (especialmente en Java, XML, PHP y Javascript) tanto del investigador como de todo aquel que se tome el tiempo de leer la presente tesis.

1.7 Metodología

1.7.1 Metodología de la Investigación

En cuanto a la metodología usada, no se encontró a un autor específico que hable sobre tipos, diseños, enfoques y métodos de una investigación, sin embargo al buscar por separado si es posible encontrar los términos. En la presente tésis se utilizaron dos libros como fuentes, el primero titulado “Tipos, métodos y estrategias de investigación. Pensamiento y Acción” (Tam, Vera y Oliveros. 2008), y el segundo se titula: “Metodología de la Investigación” (Hernández, Fernández y Baptista. 2010).

En cuanto al tipo de investigación; se puede ver el punto 3.1.1; donde se explica por qué corresponde al tipo de investigación aplicada.

Sobre el diseño de la investigación, se puede ver el punto 3.1.2; donde se explica por qué el diseño es de tipo no experimental del tipo investigación-acción.

El enfoque de la presente investigación es cualitativo, lo cual se explica más detalladamente en el punto 3.1.4.

Finalmente, el método de investigación es Experimental Pre-Experimental, lo cual se explica más detalladamente en el punto 3.1.5.

1.8 Matriz de Consistencia

En la Matriz de Consistencia de este proyecto se indican los objetivos en base a los problemas encontrados, se plantea una hipótesis general, se indican las variables identificadas y también la metodología a aplicar.

La Matriz de Consistencia de la presente tesis se muestra en la tabla 1.

Tabla 1

Matriz de consistencia del la tesis

PROBLEMA	OBJETIVOS	HIPÓTESIS	VARIABLES E INDICADORES	METODOLOGÍA
Problema Principal ¿Cómo obtener rápidamente información sobre las rutas del transporte público dentro de la ciudad del Cusco? Problemas Secundarios **P1:** ¿Cómo mantener la base de datos de manera sencilla? **P2:** ¿Cómo comprobar que las rutas ingresadas y los paraderos que se utilizan han sido correctamente ingresados? **P3:** ¿Cómo puede el usuario ingresar desde dónde y hasta dónde va a viajar? **P4:** ¿Dónde puede visualizar el usuario las empresas y rutas registradas? **P5:** ¿Cómo indicar al usuario lo que debe de hacer para transportarse haciendo uso del sistema de transporte público de la ciudad del Cusco? **P6:** ¿Cómo puede saber el usuario el camino para subir al bus, el que este va a seguir, y el camino que debe seguir al bajar del mismo? **P7:** ¿Cómo puede saber el usuario el lugar donde se encuentra mientras hace uso del aplicativo?	Objetivo General: Desarrollar una aplicación para consultas de rutas del transporte público en la ciudad del Cusco. Objetivos específicos **O1**: Desarrollar un sistema backend basado en web para dar mantenimiento a las tablas de paraderos, rutas, empresas, coordenadas de ruta (de ida y vuelta) y paraderos de ruta (ida y vuelta). **O2**: Mostrar en el sistema backend mediante consultas a la base de datos y uso de los mapas de Google; los gráficos de las rutas ingresadas para poder probarlas, así como también los paraderos que son utilizados por las rutas. **O3**: Desarrolllar una aplicación para Android donde el usuario pueda indicar los puntos de origen y destino de su viaje. **O4**: Mostrar en la app Android la ubicación del usuario, el listado de empresas y rutas registradas, y el gráfico de las rutas de ida y vuelta de las empresas. **O5**: Sugerir empresas al usuario mediante la app para que este llegue a su destino, dependiendo de los puntos de origen y destino ingresados. En caso la distancia sea muy corta entre ambos puntos o no existan rutas para cubrir esos puntos, se debe sugerir ir a pie y graficar la ruta a seguir. **O6**: Graficar, en la app, la ruta que el usuario seguirá en el bus, además del camino desde el punto de origen al paradero de subida, y el camino entre el paradero de bajada hasta el punto de destino. **O7**: Realizar, dentro de la app, seguimiento en tiempo real de la ubicación del usuario y mostrárselo para que este pueda ubicarse en todo momento durante su viaje; y baje en el lugar correcto.	El desarrollo de una aplicación móvil para consultas de rutas del transporte público de la ciudad del Cusco, brindará a sus usuarios de manera rápida la información necesaria para saber qué empresa tomar desde y hasta qué lugar, para así poder movilizarse dentro de la ciudad haciendo uso del servicio de transporte público.	1.- Variable Independiente Aplicación móvil a desarrollar Indicadores - Número de rutas ingresadas - Número de paraderos registrados. - Número de empresas registradas 2.- Variable dependiente Satisfacción del cliente - Agrado del diseño de la interfaz - Sencillez de uso. - Utilidad de la app.	Tipo de investigación: Aplicada Diseño de la investigación: Investigación-acción, Alcance: Los 8 distritos de la provincia del Cusco. Personas. Usuarios de móviles con Android nacionales o extranjeros que hablen Español o Inglés; entre los 17 y 65 años. Enfoque: Cualitativo Método: Experimental pre-experimental

Nota: Fuente propia

CAPÍTULO 2 – MARCO TEÓRICO DE LA TESIS

En este capítulo se detalla los distintos elementos utilizados en la presente tesis, así como también los conceptos que se necesita entender para comprender el contenido de la misma.

2.1 Aspectos Teóricos Pertinentes

2.1.1 Hardware: La Real Academia de la Lengua Española (RAE) define a "hardware" como *"Conjunto de aparatos constituido por una computadora y sus periféricos"*, también como: *"Conjunto de aparatos de una computadora*" (Hardware). Ampliando ambas definiciones, el "hardware" es todo componente físico de un equipo informático, empezando desde los dispositivos de entrada como el teclado, una pantalla táctil o el mouse, y terminando en los dispositivos de salida como, valga la redundancia, la pantalla, la impresora o los parlantes. También engloba a las memorias RAM, discos duros, tarjetas gráficas, de red, fuentes de alimentación, etc. (Fuente propia).

2.1.2 Software: La RAE define a "software" como el *"Conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora"* (Software). En otras palabras, es todo componente lógico que cumple una función dentro de un equipo electrónico. El software no solo se encuentra en los discos duros de una computadora y se ejecuta cuando esta lo requiere, sino también en los distintos chips ubicados en prácticamente cualquier electrodoméstico, auto, sensor, etc. Mediante una secuencia de instrucciones; el software genera los resultados esperados haciendo uso del hardware disponible. Entre los más usados figuran Windows (en sus distintas versiones), Linux, Whatsapp, Facebook, Microsoft Office, Adobe Reader, etc. (Fuente propia).

2.1.3 Sistema Operativo: Un sistema operativo (de ahora en adelante SO), es un software que inicia al enceder un equipo informático, y se hace cargo de

todos los recursos, tanto de software y hardware, que este posee, logrando que la interacción del hombre y la máquina sea mucho más sencilla. (Sistemas Operativos).

Existen muchos SO, algunos de ellos son libres y otros de pago, en nuestro medio (la ciudad del Cusco), el más conocido es sin lugar a dudas Windows, seguido y tal vez hasta superado por Android. (Fuente propia).

2.1.4 API: Es un término ampliamente usado en la informática, en inglés significa “Application Programming Interface”; y en español: “Interfaz de Programación de Aplicaciones”. Una API es un conjunto de funciones que permite la comunicación entre dos aplicaciones de manera correcta y transparente. Entre APIs muy conocidas se puede listar a los mapas de Google, los Servicios Web de Amazon, de Youtube, Facebook e incluso Flickr. (API. 2007)

2.1.5 Android: Es un SO construido sobre el kernel de linux 2.6. Fué lanzado bajo licencia Apache y con el paso de los años ha despertado el mercado de consumidores de aplicaciones móviles. Ofrece un enfoque unificado para el desarrollo de aplicaciones y actualmente es el SO de millones de dispositivos móviles en el mundo. (Nolasco J. págs 19-21).

En su libro, Nolasco nos explica que Android apareció por primera vez el 23 de Septiembre del 2008, y nos da a conocer las fechas de lanzamiento y algunos detalles de cada versión hasta el año 2011, los cuales se pueden apreciar en la tabla 2.

Tabla 2

Versiones de Android

Versión	Detalle
1.0 Apple pie	Apareció por primera vez el 23 de Septiembre del año 2008 en un dispositivo HTC Dream G1.
1.1 Banana Bread	Actualización lanzada el 9 de Febrero del 2009 para solamente T-Mobile G1.
1.5 Cupcake	Lanzada el 30 de Abril del 2009. Es la primera versión basada en el kernel 2.6.27 de Linux, y también la primera actualización oficial de Android.
1.6 Donut	Lanzada el 15 de Septiembre del 2009. Se basa en la versión 2.6.29 del kernel de Linux.
2.0 Eclair	Lanzada el 26 de Octubre del 2009. El 3 de Diciembre del 2009 se lanza la versión 2.0.1. El 12 de Enero del 2010 se lanza la versión 2.1.
2.2 Froyo	Lanzado el 20 de Mayo de 2010. Fué la primera versión capaz de brindar internet a otros equipos.
2.3 Gingerbread	Lanzado el 6 de Diciembre de 2010. Implementa las llamadas por internet. El 22 de Febrero del 2011 se lanza la versión 2.3.3.
3.0 Honeycomb	También el 22 de Febrero del 2011 se lanza la versión para tablets. La versión 3.1 es lanzada el 10 de Mayo del mismo año. Especialmente optimizado para dispositivos con pantallas de mayor tamaño. El 18 de Julio del 2011 se lanza la versión 3.2.
4.0 Ice Cream Sandwich	Incluye la nueva fuente de texto (hoy ampliamente usada) llamada Roboto. Soporte para wifi directo.

4.1 Jelly Bean	Una de las versiones mas recordadas hoy en día, lanzada el 24 de Julio del 2012. Incluye las versiones 4.2 y 4.3.
4.4 KitKat	Implementa software de reconocimiento de voz, realizando así algunas tareas con solo la voz del usuario. (Android 4.4 KitKat).
5.0 Lollipop	Incluye a los relojes, televisores y autos como elementos donde Android puede ser instalado. Incluye Material Design. (Android 5.0 Lollipop)
6.0 Marshmallow	Permite al usuario tener control sobre los permisos que otorga a aplicaciones. Mejor uso de la batería, generando un mayor tiempo de independencia al dispositivo. (Android 6.0 Marshmallow)
7.0 Nougat	Lanzado el 17 de Abril del 2017. Algunos de sus beneficios son: - Permite usar mas de un idioma al escribir. - Vista de ventanas múltiples. - Modo de realidad virtual. - Encriptación a nivel de los archivos. (Android 7.0 Nougat)
8.0 Oreo	Lanzado el 21 de Agosto del 2017. Es la versión más nueva a la fecha. Algunos de sus beneficios son: - Mayor velocidad - Gasto de batería menor. - Incluirá una version llamada Android Go, para dispositivos de gama baja

Nota: Basado en el libro de Nolasco J. Y los artículos encontrados en "The Android Story"

2.1.6 Software Libre: De acuerdo a la revista Users (Ubuntu, Instale y domine el sistema Linux … 2012), el software libre se define como: *"Principalmente, una modalidad de desarrollo y distribución de programas de computadoras"*, (pág 12) y pocas líneas mas abajo amplía la definición a: *""El software libre*

es un suceso social que esta cambiando la forma en que el mundo informático se mueve … Como definición de software libre, decimos que es aquel que se puede utilizar y distribuir libremente. También puede ser modificado y vuelto a distribuir (para estas últimas opciones requiere código abierto). Y para que el programa sea considerado Software Libre, deben garantizarse esas cuatro libertades. Si la licencia de uso y distribución del programa no garantiza alguna de las cuatro libertades mencionadas, entonces, nos encontraremos ante un programa que no es libre) (pág 12).

2.1.7 Linux: Vale la pena aclarar que usualmente las personas se refieren a Linux como un SO, sin embargo en sus orígenes no fue así, originalmente, Linux fue un Kernel, lo cual se entiende como las líneas de código encargadas de gestionar ese pequeño espacio entre el software y el hardware, y también ejecuta las tareas que el SO le ordena y plasmarlas en el hardware, en pocas palabras, el corazón de un SO. (¿Qué es Kernel y para qué sirve?, 2014). Este kernel (Linux) se unió con un SO que existía en ese entonces llamado GNU, formando el SO llamado GNU/Linux, el cual actualmente se conoce como Linux.

Habiendo entendido lo anterior, podemos decir que Linux es un SO, por lo tanto es un programa que permite al usuario interactuar con su computador, y que incluye una serie de programas que facilitan las tareas que un usuario realiza, y que a su vez brinda un medio donde se pueden desarrollar nuevas heramientas. (¿Qué es GNU/Linux?).

Linux no tiene dueño, lo cual es una de sus características más interesantes, pertenece a todos y es mantenido por una gran comunidad de desarrolladores, testeadores, etc. Año tras año salen a la luz nuevas distribuciones de Linux, que son básicamente el sistema base Linux; más ciertos programas seleccionados. En el presente proyecto se utiliza Ubuntu, que es una distribución de Linux basado en Debian (que es la distribución oficial de la Free Software Foundation), que ganó gran popularidad y prestigio a lo largo de los años dada la excelente gama de aplicaciones que

incluye y la facilidad que brinda a sus usuarios para realizar las tareas que necesita, además de una interfaz amigable para el proceso de instalación, la cual ayuda bastante al momento de iniciarse en el mismo. (Ubuntu, Instale y domine el sistema Linux …, 2012, pág 17-20).

2.1.8 Metodologías de Desarrollo de Software: Es bueno entender primero qué es una metodología; según la RAE (Real Academia de la Lengua Española), una metodología es un *"Conjunto de métodos que se siguen en una investigación científica o en una exposición doctrinal"* (Metodología). En el mundo de informática, una metodología mantiene en el núcleo la definición de la RAE, sin embargo engloba algunas cosas extra.

Una metodología de desarrollo de software es un conjunto de procedimientos, técnicas e indicaciones para hacer documentación; que sirven para lograr un software de calidad, es similar a una receta de cocina donde se va indicando paso a paso lo que se debe de hacer a continuación para lograr el resultado esperado, dejando en claro las personas que deben participar, los procesos que deben seguirse, los documentos que deben realizarse, entre otras cosas. (Metodologías de Desarrollo de Software)

En el caso particular de este proyecto, se utiliza una mezcla de PUDS (ver 2.1.10) y Scrum (ver 2.1.12), el primero es ampliamente aceptado como una metodología, y el segundo es un marco de trabajo que actualmente gana más y más terreno en el ambiente del desarrollo de software.

2.1.9 UML (Unified Modeling Language): El Lenguaje de Modelado Unificado fué creado en la década de los 90 por los "tres amigos" (Booch, Rumbaugh y Jacobson) con la finalidad de unificar la representación de los sistemas de manera estandarizada. UML ayuda a representar las distintas etapas del desarrollo de software por medio de diagramas (a modo de planos), representando etapas, requisitos, actividades, etc; de manera simple y eficiente, centrándose en los requisitos funcionales que el software tiene y los procesos de negocio del mismo. (Coronel E. 2010. Pág 32).

Por otro lado si tomamos en cuenta el texto de Ferré J. & Sánchez I. (s.f.), ampliamos la definición de UML un poco más, *"Es un lenguaje que permite modelar, construir y documentar los elementos que forman parte de un sistema orientado a objetos"* (pág. 1) Vale la pena destacar que el principal objetivo de UML al momento de ser creado fue de posibilitar el intercambio de modelos entre las distintas herramientas CASE (Computer Aided Software Engineering, en español: Ingeniería de Software Ayudado por Computadoras) (pág. 2)

UML es un lenguaje relativamente amplio, y en el presente proyecto no haremos uso del 100% de este lenguaje, sino de lo necesario; ya que se seguirá los consejos del PUDS junto a SCRUM. Dicho esto, es importante entender algunos de los elementos más importantes que engloba UML, como por ejemplo:

- Notas: Son usadas para añadir información adicional a cualquier elemento de un diagrama (Ferré J. & Sánchez I. s.f. pág. 3).

- Caso de Uso: Un caso de uso (CU) es un requisito del sistema, una necesidad que tienen los usuarios del sistema (actores, tanto humanos como otros sistemas) sobre el mismo. Especifican comportamientos Pueden ser funcionales como no funcionales, los primeros agregan valor y son imprescindibles, y los segundos son básicamente restricciones.

- Actor: Un actor es un usuario, independiente de si es humano o no, que hace uso de alguna función del sistema. Suelen corresponderse con trabajadores, y cada uno de ellos cumplen una o más funciones, las cuales se representan como casos de uso. (Jacobson et al., 1999, pág. 128-129). Los autores también dividen a los trabajadores según su responsabilidad sobre los artefactos en: analista de sistemas, especificador de casos de uso, diseñador de interfaces de usuario y arquitecto. (pág.132-134)

- Relaciones: Existen varios tipos de relaciones entre elementos, entre ellos podemos destacar:
  - Dependencia: Indica que el cambio en el elemento destino puede implicar un cambio en el elemento origen. Se representa por una línea de trazo discontinuo, el elemento ubicado en el origen de la fecha es el dependiente, por lo que se entiende que la Clase 1 depende de la Clase 2. Esto se representa en la figura nro 1 (Ferré J. & Sánchez I. s.f. pág. 3-4).

Figura 1. Representación de la relación de dependencia entre clases

  - Asociación: Sirven para representar un enlace, el cual es una conexión entre objetos. Pueden ser bidireccionales aunque algunas veces son en un solo sentido. Se representan por una línea continua y una flecha al final; como se ve en la figura 2. Dependiendo del caso, pueden tener multiplicidad. (Berzal F. s.f. pág. 24)

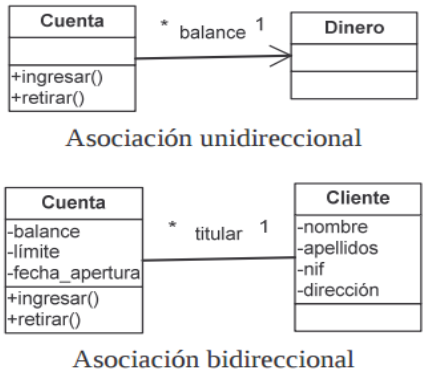

Figura 2. Relación de asociación unidireccional y bidireccional.

  - Generalización: También conocido como Herencia. *"Objetos de distintas clases pueden tener atributos similares y exhibir comportamientos parecidos"* (Berzal F. s.f. pág. 31). Se representa

(ver figura 3) por una línea continua y un triángulo vacío que actúa como flecha al final. El origen de la flecha es el que hereda, y la punta es el 'padre'.

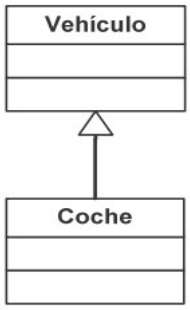

Figura 3. Gráfico de la relación de dependencia entre clases.

- Clases de análisis: *"Representa una abstracción de una o varias clases y/o subsistemas del diseño del sistema. Esta abstracción posee las siguientes características:*
  - *Se centra en el tratamiento de los requisitos funcionales y pospone los no funcionales.*
  - *Raramente define y ofrece una interfaz en términos de operaciones y de sus signaturas. En cambio, su comportamiento se define mediante responsabilidades en un nivel más alto y menos formal.*
  - *Define atributos normalmente conceptuales y reconocibles en el dominio del problema.*
  - *Participa en relaciones.*
  - *Siempre encajan en uno de tres estereotipos básicos, de interfaz, de control o de entidad."* (Jacobson et al., 1999, pág. 173)

- Estereotipos: Son tres los estereotipos estandarizados por UML, interfaz, entidad y control. El de interfaz se utiliza para modelar la interacción entre el sistema y sus actores, generalmente representan abstracciones de ventanas, formularios, sensores, terminales, etc. El de entidad se utiliza para modelar información que posee persistencia, o sea una manera de almacenar que le brinda una vida larga, suelen mostrar una

estructura de datos lógica explicando de qué información depende el sistema. Finalmente los de control representan, como su nombre lo indica, cálculos complejos, coordinaciones, secuencia, transacciones, etc. Los estereotipos se representan en la figura Nro. 4

Figura 4. Representación de los estereotipos.

El uso de UML genera los distintos modelos que se utilizan para mostrar diferentes diseños o etapas en el proceso de desarrollo de software, entre estos modelos tenemos:

2.1.9.1 Modelo de Casos de Uso: Representa los requisitos que tiene un sistema de software, además de las relaciones que tienen estos con los actores. Sirve como acuerdo entre el cliente y los desarrolladores, y contiene, valga la redundancia, casos de uso, actores y sus relaciones. UML permite presentar el diagrama en diferentes partes dependiendo del tamaño que tenga el mismo. (Jacobson et al., 1999, pág. 127-128)

2.1.9.2 Modelo de Clases: Las clases son el núcleo de un sistema orientado a objetos, y este diagrama busca mostrar las clases existentes en un sistema y las relaciones entre ellas. (Diagramas de Clases en UML)

2.1.9.3 Diagrama de Actividades: Sirven básicamente para modelar el flujo que existe entre actividades. Se basa en arcos y nodos, y muestra cómo fluye el control entre clases con el objetivo de lograr un fin. Contiene básicamente estados de actividad, estados de acción, transiciones y objetos. (Ferré J. & Sánchez I. s.f. pág.14).

2.1.10 Proceso Unificado de Desarrollo de Software (PUDS): Si bien el PUDS es considerado como una metodología, lo cierto es que sus creadores lo denominan literalmente de la siguiente manera:

"El Proceso Unificado es un proceso de desarrollo de software. Un proceso de desarrollo de software es el conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema software. Sin embargo, el Proceso Unificado es más que un simple proceso; es un marco de trabajo genérico, que puede especializarse para una gran variedad de sistemas software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes tipos de aptitud y diferentes tamaños de proyectos" (Jacobson I., Booch G. y Rumbaugh J. 1999, pág. 4)

El PUDS utiliza a UML como un lenguaje estándar para mostrar los diferentes esquemas que utiliza, sin embargo definir PUDS no estaría completo sin indicar que se resume en tres frases muy importantes:

- Dirigido por casos de uso:

  Los casos de uso representan requisitos funcionales de un sistema a desarrollar, y son el punto de partida para todo el proceso de desarrollo del mismo. (Jacobson et al., 1999, pág. 5) La captura de los casos de uso es un proceso crucial en el desarrollo de sistemas, es de suma importancia tener los casos de uso funcionales completamente entendidos y descritos, ya que son los que más valor agregan al sistema.

  Cada usuario en un sistema se representa como un actor y los requisitos como casos de uso, y los modelos de casos de uso representan la interacción de los actores con todos los casos de uso (Jacobson et al., 1999, pág. 33), sin embargo; ¿Por qué usar casos de uso?, los autores lo definen de manera muy simple: *""Proporcionan un medio sistemático e intuitivo de capturar requisitos funcionales centrándose en el valor añadido para el usuario. Dirigen todo el proceso de desarrollo debido a que la mayoría de las actividades como el análisis, diseño, y prueba se*

llevan a cabo partiendo de los casos de uso. El diseño y la prueba pueden también planificarse y coordinarse en términos de casos de uso. Esta característica es aún más evidente cuando la arquitectura se ha estabilizado en el proyecto, después del primer conjunto de iteraciones" (Jacobson et al., 1999, pág. 35)

- Centrado en la arquitectura:

  La arquitectura en un software es similar a la arquitectura convencional usada para la construcción, ya que muestra un diseño del software a realizar centrado en los casos de uso clave; y en el software y hardware para el que se implementará, (Jacobson et al., 1999, pág. 5-6) además, no se debe olvidar que los casos de uso no son suficientes para desarrollar un software, sin la arquitectura no se puede tener una visión global y más completa del mismo.

  La arquitectura de software, como se dijo antes, es similar a la de la construcción, integra las distintas interfaces necesarias con todos los elementos que dan soporte, en otras palabras, todo lo que ve y no ve el usuario final. De acuerdo a los autores, va condicionada por muchos factores, entre los que se encuentran: el o los sistemas operativos sobre los que se usará, sistemas de gestión de bases de datos, productos "middleware" como marcos de trabajo o sistemas existentes para. por ejemplo; el envío de correos, sistemas heredados, políticas de la empresa o el país, requisitos no funcionales como el uso de memoria o la disponibilidad del mismo, distribución del mismo, etc. (Jacobson et al., 1999, pág. 62)

- Iterativo e incremental:

  Un proyecto se divide en miniproyectos, siendo cada uno de estos una iteración que resulta en un incremento, cada iteración se basa en un grupo de casos de uso; para luego basarse en los artefactos que se va generando. (Jacobson et al., 1999, pág. 6-8)

  Entre los beneficios de un proceso con iteraciones controladas tenemos:

- Reduce el riesgo a los costes de un solo incremento.
- Reduce el riesgo de no sacar el producto en el tiempo programado.
- Acelera la velocidad de desarrollo.
- Reconoce que los requisitos que se generan al inicio no son todos los que existirán en el proyecto, ya que a medida que se avanza estos se van refinando y redefiniendo.

Las primeras iteraciones dejan en claro elementos como la viabilidad del sistema y los posibles riesgos, de esa forma tanto los casos de uso como la arquitectura, se van afianzando más y más. Es importante también tener en cuenta lo que es y lo que no es una iteración, como se dijo antes, una iteración es un proceso que agrega valor y cumple un objetivo, apoya con la gestión de cambios y permite adecuarnos a las nuevas necesidades que aparecen luego de un proceso de refinamiento. De acuerdo a los autores; una iteración no es: *"un desarrollo aleatorio, un parque de juegos para los desarrolladores, algo que afecte solo a los desarrolladores, rediseñar una y otra vez lo mismo hasta que al final los desarrolladores prueben algo que funciona, algo impredecible, una excusa para fracasar en la planificación y en la gestión"* . (Jacobson et al., 1999, pág. 84-85)

El proceso unificado se repite en ciclos, cada ciclo concluye una versión del producto. Cada ciclo consta de cuatro fases; inicio, elaboración, construcción y transición, y cada una de estas fases consta de un número de iteraciones (pág. 10). Esto se aprecia mejor en la figura 5.

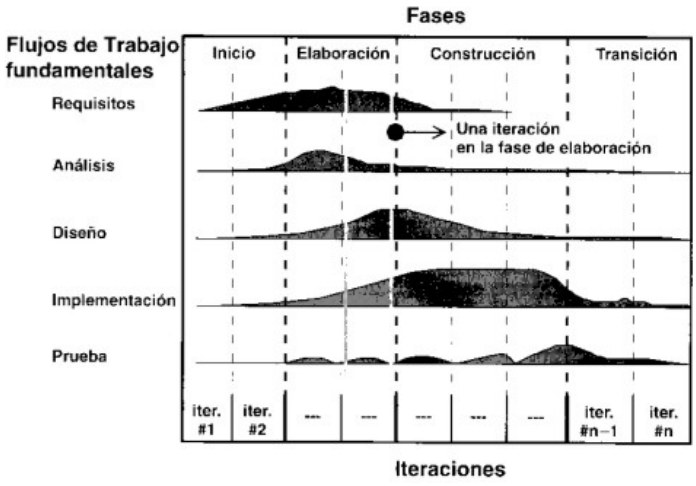

Figura 5. Fases y Flujos de trabajo del Proceso Unificado. Fuente (Jacobson I., Booch G. y Rumbaugh J. 1999)

El producto que se va desarrollando a medida que los ciclos van avanzando, debe ser; como se puso anteriormente; un producto terminado, que sea representativo no solamente para el usuario final, sino para los desarrollares e interesados en el software. Los creadores del PUDS, (Jacobson I., Booch G. y Rumbaugh J. 1999, pág.10) indican textualmente lo que debe acompañar a cada ciclo:

- *"Un modelo de casos de uso, con todos los casos de uso y su relación con los usuarios.*
- *Un modelo de análisis, con dos propósitos: refinar los casos de uso con más detalle y establecer la asignación inicial de funcionalidad del sistema a un conjunto de objetos que proporcionan el comportamiento.*
- *Un modelo de diseño que define: (a) la estructura estática del sistema en forma de subsistemas, clases e interfaces y (b) los casos de uso reflejados como colaboraciones entre los subsistemas, clases e interfaces.*
- *Un modelo de implementación, que incluye componentes (que representan al código fuente) y la correspondencia de las clases con los componentes.*
- *Un modelo de despliegue, que define los nodos físicos (ordenadores) y la correspondencia de los componentes con estos nodos.*

- *Un modelo de prueba, que describe los casos de prueba que verifican los casos de uso.*
- *Y por supuesto, una representación de la arquitectura."*

Las 4 "P" en el desarrollo de software es otro punto a resaltar en el PUDS (los autores incluyen "herramientas" como un quinto elemento), estos son: (Jacobson et al., 1999, pág. 13-28)

- Personas: Son los arquitectos, desarrolladores, programadores, clientes, ingenieros de pruebas, clientes, etc. Se refiere directamente a personas humanas. Un detalle importante en PUDS es que a diferencia de UML, el término "roles" no es usado, en cambio se usa la palabra "trabajadores", esto para dejar en claro que un trabajador (una persona) puede desempeñar roles distintos en diferentes etapas del proyecto. (Jacobson et al., 1999, pág. 16).

- Proyecto: *"Elemento organizativo a través del cual se gestiona el desarrollo de software"* (Jacobson et al., 1999, pág. 13).
  Al hablar de proyecto es muy importante considerar que primero; es una secuencia de cambio, ya que cada ciclo genera una nueva versión, pero el camino para obtener los productos son los cambios. Segundo, es una serie de iteraciones; ya que dentro de cada fase se lleva a cabo diferentes iteraciones (requisitos, diseño, implementación y prueba) para lograr los resultados, y tercero; es un patrón organizativo, ya que los trabajadores van cumpliendo los distintos roles que les corresponden siguiendo ciertos patrones para lograr las metas y a su vez generar los artefactos necesarios. (Jacobson et al., 1999, pág. 18)

- Producto: Artefactos que se van generando a medida que se desarrolla el proyecto. (modelos, ejecutables, codigo fuente, etc). Por lo tanto, el producto no solo es el software final que se entrega, sino también el sistema entero.

- Artefactos: Es *"Cualquier tipo de información creada, producida, cambiada o utilizada por los trabajadores en el desarrollo del sistema"* (Jacobson et al., 1999, pág. 18). Existen dos tipos; artefactos de ingeniería y artefactos de gestión.
- Colección de modelos: Vale la pena dejar en claro que un modelo es una abstracción del sistema (el aislamiento de un elemento particular de un sistema), y que construir un sistema significa diagramar y construir una serie de modelos, los cuales representan pequeñas porciones del producto en distintos ciclos. La idea de un modelo es que sea autocontenido, o sea que no necesite de modelos externos para que se entienda lo que representa. (Jacobson et al., 1999, pág. 19-21)

- Proceso: Son las distintas definiciones de conjuntos de actividades que se desarrollan para transformar requisitos en productos. Para el PUDS un proceso es una plantilla de cómo debe llevarse a cabo un conjunto de actividades; en otros términos, *"Un proceso del desarrollo de software es una definición del conjunto completo de actividades necesarias para convertir los requisitos de usuario en un conjunto consisten de artefactos que conforman un nuevo producto software"* (Jacobson et al., 1999, pág. 22). Los procesos varían dependiendo de a los distintos contextos; ninguno es de aplicabilidad universal, y los factores principales que influyen en cómo se diferenciará el proceso son: (Jacobson et al., 1999, pág. 24)
  - Factores organizativos: Software existente, la estructura de la empresa, la cultura organizativa, etc.
  - Factores del dominio: *"El dominio de la aplicación, procesos de negocio que se deben soportar, la comunidad de usuarios y las ofertas disponibles de la competencia."*
  - Factores del ciclo de vida: Tiempo en el que saldrá al mercado, tiempo de vida esperado del software, etc.
  - Factores técnicos: Lenguajes de programación, herramientas usadas, bases de datos, etc.

- Herramientas: Es el software utilizado para automatizar los procesos. Dicho esto, se puede concluir sin temor a equivocarse que las herramientas son imprescindibles al momento de desarrollar software, tanto para crear, dar mantenimiento, generar persistencia de datos u otras actividades de carácter obligatorio en el desarrollo de software; además de realizar un sinfín de tareas repetitivas. Las herramientas deben ser de fácil uso, y en lo posible deben de proporcionar a los trabajadores las mejores opciones para reutilizar todo lo que sea posible. (Jacobson et al., 1999, pág. 25-29)

Finalmente, como se vió en la figura 5, las fases del proceso de desarrollo son 4, y se definen de la siguiente manera:

- **Fase de Inicio:** Establece el análisis del negocio y es la etapa en la que se decide si seguir o no con el proyecto. Indica los límites del sistema, se propone la arquitectura del sistema, identifica riesgos críticos e incluso se puede elaborar un prototipo para mostrar a los potenciales clientes (pág. 305-306). Durante la fase de inicio también se debe generar una propuesta de arquitectura, detallar los casos de uso pertinentes que nos lleven a una mejor comprensión de lo que se desea obtener, y por supuesto, realizar un estimado del coste que tendrá el proyecto.

  Entre los productos que se obtienen después de completar la fase de inicio, podemos resaltar los siguientes (pág. 343-344)
  - Lista de características.
  - Primera versión del modelo de negocios.
  - Arquitectura candidata.
  - Lista inicial de riesgos y mitigación de los mismos.

- **Fase de Elaboración:** Genera la arquitectura sobre la cual se desarrollará el sistema, identifica riesgos significativos, recopila los casos de uso funcionales aproximadamente al 80%, y prepara un listado de costes (pág. 306-307). Además, en esta etapa se completa la descripción de los casos de uso y la prioridad de los mismos, se realizan

prototipos de interfaces de usuarios, se determina la arquitectura que se va a utilizar, se prepara la propuesta económica y la manera en que se va a recuperar la inversión, y se planifica la etapa de construcción. (pág. 345-365). Entre los productos clave que se obtienen en esta etapa podemos destacar: (pág. 365)

- Modelo de Negocios.
- Propuesta económica.
- Lista de riesgos actualizada.
- Arquitectura.

- **Fase de Construcción:** Es la etapa más grande, es donde se genera el producto de software el cual es probado y corregido. Incluye la identificación, descripción y realización de todos los casos de uso, la monitorización y mitigación de todos los riesgos que se hayan podido presentar, modifica la arquitectura de ser necesario. (pág. 307-308)

- **Fase de Transición:** Es la etapa donde se entrega el producto a los usuarios. Incluye la preparación para la entrega, manuales de uso, ajustes finales para integrar el software al entorno actual, sugerencias sobre actualizaciones, y también registra y evalúa las lecciones aprendidas para referencia futura; además de registrar elementos útiles para una siguiente versión. (pág. 308)

  Durante el proceso de transición, existen ciertos pasos que los proyectos deben de seguir, entre los que podemos resaltar: (pág. 386-387)

  - Preparar la versión beta del producto.
  - Instalar esta versión en los dispositivos de usuarios seleccionados para pruebas.
  - Recaudar información a partir de las pruebas realizadas.
  - Adaptar el producto de acuerdo a las necesidades de los usuarios, implementando las nuevas ideas que se consideren posibles y necesarias

Se debe tener en cuenta que en el presente proyecto no se han elaborado todos los modelos de manera obligatoria; sino aquellos que se han considerado necesarios, ya que estamos usando PUDS como metodología y Scrum como marco de trabajo.

2.1.11 Metodologías Ágiles: Son un conjunto de métodos y, valga la redundancia, metodologías; que permiten el desarrollo de software en las distintas áreas que este conlleva; gestión, arquitectura, diseño, etc. Sin embargo, no solamente se centra en ser un conjunto de procesos, sino también una mentalidad. En una metodología ágil las decisiones no se toman de manera personal por el gerente, sino de manera que intervenga todo el equipo; creando un ambiente en donde la información sea compartida por todos y donde cada persona tiene la palabra en los procesos que se aplican. (Stellman & Greene, 2015, pág. 2).

Las metodologías ágiles también son iterativas e incrementales, pero no son iguales a las tradicionales, integran distintos artefactos y diferentes procesos que hacen que el proceso de desarrollo sea más eficiente y a su vez más veloz. El núcleo de las metodologías ágiles son cuatro valores, estos fueron acuñados en el año 2001 por un grupo de 17 personas con ideas similares. Tras conversar por varios días sobre la solución a los problemas de software que tuvieron durante sus carreras, acordaron varias ideas y principios, entre ellos el mismo término “ágil”. (pág. 33). El Manifesto Ágil (Agil Manifesto), surgió de esa reunión, y dice literalmente:

“Estamos destapando mejores formas de desarrollar por medio de hacerlo y de ayudar a otros a hacerlo. A través de este trabajo hemos llegado a valorar :

- ***Los individuos y las iteraciones*** *por encima de los procesos y las herramientas.*
- ***Software funcional*** *por encima de documentación exhaustiva.*
- ***Colaboración del cliente*** *por encima de negociaciones de contrato.*

- ***Respuesta al cambio*** *por encima de seguir un plan.*

Eso es, mientras hay valor en los items de la derecha, nosotros valoramos más los items de la izquierda" (pág. 33)

A modo de profundizar sobre estos cuatro valores, se procede a detallar un poco más cada uno de ellos:

El primero nos habla que el desarrollo de software esta lleno de buenas prácticas, pero no todas ellas son adecuadas para un proyecto en específico. También que es muy importante entender a los miembros del equipo de desarrollo, entender cómo se llevan los unos con los otros y entender cómo estos impactan en la vida de los otros. Por lo tanto, aunque se tengan excelentes procesos, si los miembros no están en la misma sintonía, estos no tomarán los procesos con la misma importancia y seriedad, incluso peor, tal vez lo tomen muy literalmente aunque esto los lleve a errores e incoherencias. Es más importante realizar un proceso coherente y entendido por todos. (pág. 34).

El segundo nos pone al tanto que existen muchísimos archivos guardados sobre documentación de software, y que vale más la pena tener software funcional que documentación exhaustiva. Sin embargo esto no significa que no se deba documentar, sino tratar de enfocarse en documentos que aportan información necesaria sin volverse problemáticos de crear y de mantener. Se debe tener en cuenta que generalmente el que escribe el código es el mismo que genera la documentación. También nos habla sobre el significado de software funcional, que en términos simples se define como software que aporte valor, ya sea beneficios económicos, mejora en el rendimiento de los empleados, y que para que un proyecto aporte valor, también debe de generar o ahorrar más dinero que lo que ha costado en realizarse. (pág. 35).

El tercer valor nos indica que el trabajo en equipo es extremadamente importante, muchas empresas (incluyendo las que desarrollan), prefieren

realizar contratos incluso internamente, para que de esa manera si algo falla; entonces ciertos grupos se libran de culpa y pueden apuntar el dedo a un culpable directo. Esta actividad, de acuerdo a este valor, es contraproducente, ya que en lugar que el equipo trate de aportar valor y mejorar el proceso, están más preocupados en protegerse a si mismos. Una de las maneras de evitar esto es tener un Product Owner (Dueño del Producto) que se integre dentro del equipo sin necesidad de desarrollar directamente y que tenga un puesto realmente importante, de modo que sienta que realmente es el dueño y que colabora con el resto del equipo. (pág. 35)

El cuarto valor pone en evidencia que es importante reaccionar a los cambios en lugar de seguir estrictamente un plan, los equipos siempre deben tratar de mejorar y modificar lo que sea necesario para cumplir de la mejor manera con las necesidades del cliente. Se debe tener en cuenta que si se sigue un plan erróneo; el resultado también será erróneo, y que el cambio no es malo, especialmente si se detecta en una etapa temprana, ya que si el cambio se realiza posteriormente, el proceso será mucho más costoso (pág. 35)

2.1.12 SCRUM: El año 1986, Hirotaka Takeuchi e Ikujiro Nonaka publicaron una reseña llamada “The New New Product Development Game” (El Nuevo Nuevo Juego de Desarrollo de Productos) en la revista “Harvard Business Review”, explicando así; cómo Scrum se aplica no solo al desarrollo de software, sino a prácticamente cualquier problema de gestión. (Takeuchi and Nonaka, The Roots of Scrum) (ver también en referencias, The New New Product Development Game).

Antes de continuar explicando lo que es Scrum, es bueno dejar en claro que la redacción de esta fracción del marco teórico sigue la estructura de “La Guía de Scrum” (Schwaber & Sutherland, 2016), por lo tanto; si bien en un comienzo puede haber términos que no se comprendan, es, después de

haber leído la guía en su totalidad, una excelente estructura para explicar lo que es Scrum y cómo funciona.

Scrum, como marco de trabajo para el desarrollo de software complejo, fué creado por Ken Schwaber y Jeff Sutherland, y vale la pena que ambos también fueron parte del grupo de 17 personas que generaron el Manifesto Ágil, (ver 2.1.8 – Metolodogías Ágiles) (Creating Scrum), Ambos también son los escritores de “La Guía de Scrum”, la cual se encuentra en cerca de 40 idiomas y detalla de manera sencilla; cómo funciona Scrum.
De acuerdo a “La Guía de Scrum” , Scrum consiste en: Los equipos Scrum, sus roles, eventos, artefactos, y las reglas que los relacionan, además, está basado en el empirismo (conocimiento basado en la experiencia) y posee un enfoque iterativo e incremental (pág. 3)

Scrum se soporta en tres pilares, estos son:

- **Transparencia:** Los aspectos significativos de los procesos deben ser totalmente visibles y entendibles para aquellos que son responsables de los resultados, además debe existir un estándar común, de modo que se comparta una definición clara de “terminado” (pág. 4)
- **Inspección:** Los usuarios Scrum deben inspeccionar los artefactos con la suficiente frecuencia para que esto no se convierta un inconveniente, y que tampoco se pase por alto variaciones indeseadas en el camino al objetivo. (pág. 4)
- **Adaptación:** Si se detecta que en el proceso de desarrollo de software surgen aspectos que desvían el proceso de los límites aceptables, se debe realizar reajustes para minimizar estas desviaciones, para ello Scrum cuenta con: Planificación del Sprint, Scrum diario, revisión del sprint y la retrospectiva del sprint. (pág. 4)

Scrum también engloba un conjunto de valores, los cuales son:

- Compromiso.
- Coraje.
- Foco.

- Apertura.
- Respeto.

Para ampliar un poco el significado de estos 5 valores, la “Guía de Scrum” dice textualmente:

El uso exitoso de Scrum depende de que las personas lleguen a ser más virtuosas en la convivencia con estos cinco valores. Las personas se comprometen de manera individual a alcanzar las metas del Equipo Scrum. Los miembros del Equipo Scrum tienen coraje para hacer las cosas y trabajar en los problemas difíciles. Todos se enfocan en el trabajo del Sprint y en las metas del equipo Scrum. El Equipo Scrum y sus interesados acuerdan a estar abiertos a todo el trabajo y a los desafíos que se les presenten al realizar su trabajo. Los miembros del Equipo Scrum se respetan entre sí para ser personas capaces e independientes.” (pág. 4-5)

Es necesario también entender como se divide un Equipo Scrum, los eventos que este marco de trabajo contiene, y los artefactos que utiliza, para ello empecemos con el equipo Scrum.

El Equipo Scrum (Scrum Team)

Un detalle muy importante que se debe recalcar en los equipos Scrum es que son autoorganizados y multifuncionales, esto quiere decir que eligen la mejor manera de llevar a cabo su trabajo sin ser dirigidas por personas externas al equipo, y que tienen todas las competencias necesarias para poder cumplir con el trabajo sin necesidad de depender de nadie fuera del equipo. Además, entregan software de manera iterativa e incremental, asegurandose que siempre se entrega software terminado y funcional (pág. 5).

Un Equipo Scrum están conformado idealmente por:

- **Dueño del Producto (Product Owner):** Sus funciones de acuerdo a la Guía de Scrum son: (pág. 5-6)
  - ✔ Maximizar el valor del producto y el trabajo del Equipo de Desarrollo.
  - ✔ Único(a) responsable de gestionar la Lista del Producto (Product Backlog). Puede delegar algunas funciones al Equipo de Desarrollo, pero sigue siendo el responsable.
  - ✔ Es una única persona, no un comité, sin embargo puede representar a varias personas.
  - ✔ Toda la organización respeta sus decisiones, y esto se refleja en el contenido y la prioridad de la Lista de Producto.

- **Equipo de Desarrollo (Development Team):** Consiste en un grupo de personas encargadas de entregar un producto terminado al final de cada Sprint, este producto tiene como característica el lo suficientemente bueno y completo como para poner en producción. Estos tienen las siguientes características: (pág. 6)
  - ✔ Son autoorganizados. Este es un punto muy importante para los Equipos de Desarrollo, textualmente la guía nos indica: *“”Nadie (nisiquiera el Scrum Master) indica el Equipo de Desarrollo cómo convertir en elementos de la Lista de Producto en Incrementos de Funcionalidad potencialmente desplegables”* (pág. 6)
  - ✔ Son multifuncionales. Esto quiere decir que cuentan con los distintos perfiles de personas necesarios para crear Incrementos de Producto.
  - ✔ Scrum no reconoce títulos en el Equipo, todos son desarrolladores, independientemente del rol que desempeñen.
  - ✔ Scrum no reconoce subequipos en los Equipos de Desarrollo.
  - ✔ Sus miembros, al ser multifuncional, tienen distintos perfiles y especializaciones, sin embargo la responsabilidad recae en todo el equipo; como una unidad.
  - ✔ Se recomienda que el número de miembros sea de al menos 3 personas, y no menor a 9.

- **Scrum Master:** En español podría traducirse en “Maestro Scrum”, sin embargo nisiquiera en la versión en español de la guía de Scrum se traduce este término, dicho esto procedemos a explicar qué es y cuáles son sus funciones.

  El Scrum Master es el responsable que Scrum se adopte en la organización, es un líder al servicio del equipo Scrum, y ayuda a las personas externas al Equipo a entender qué cosas pueden ser útiles y cuáles no para el proyecto. (pág. 7)

  Da servicio al Dueño del Producto en varias formas, como por ejemplo:

  ✔ Encontrar técnicas de gestión efectiva de la Lista de Producto, y asegurarse que el Dueño conozca cómo ordenar la Lista de manera que maximice el valor del producto.
  ✔ Ayuda al Equipo Scrum a tener claros y concisos los elementos de la Lista de Producto.
  ✔ Entenderla planificación del producto en un entorno empírico.
  ✔ Facilitar los eventos de Scrum según se requiera o necesite.

  Da servicio al Equipo de Desarrollo de las siguientes maneras:

  ✔ Guiarlos en ser autoorganizados y multifuncionales.
  ✔ Ayuda al equipo a crear productos de alto valor.
  ✔ Eliminar impedimentos para el desarrollo del equipo.
  ✔ Facilitar los eventos Scrum según se requiera o necesite.
  ✔ Guiar a Equipos de Desarrollo en entornos donde Scrum no se conoce o no ha sido adoptado por completo.

  Los servicios que brinda el Scrum Master también se extiende a la organización de la siguiente manera:

  ✔ Liderar y guiar en la adopción de Scrum.
  ✔ Planificar las implementaciones de Scrum.
  ✔ Ayudar y guiar a los empleados en la adopción de Scrum y el desarrollo empírico de productos.

- ✔ Trabajar con otros Scrum Masters para mejorar la aplicación de Scrum en toda la organización.

Los Eventos de Scrum: Así como Scrum posee Equipos Scrum, también posee eventos, y es importante entenderlos para comprender este marco de trabajo. Para empezar a comprender los eventos, tengamos en claro que estos son lapsos de tiempo, (time boxes), de modo que estos tienen un tiempo de duración máxima. Por cierto, la referencia que se utiliza para esta parte del marco teórico es la versión en Español de la Guía de Scrum, sin embargo se mantiene la terminología en Ingles para evitar posibles confusiones. Los eventos son: (pág. 8)

- **El Sprint:** Es el corazón de Scrum, su punto esencial. Es un bloque de tiempo (time box) de un mes o menos; durante el cual se desarrolla un incremento de producto terminado; potencialmente desplegable. Tienen un objetivo, lo cual se entiende sencillamente en que se desarrollan para lograr algo, se recomienda que su duración sea consistente a lo largo del proyecto, y ni bien termina uno, comienza inmediatamente el siguiente.

  Un Sprint consiste en los siguientes eventos.

- **Planificación del Sprint (Sprint Planning).**
  Es el tiempo en el que se planifica el trabajo que se va a realizar durante el Sprint a planificar. Tienen un máximo de 8 horas para un Sprint de un mes.

  Obedece a dos preguntas:
  - ✔ ¿Qué puede hacerse en este Sprint?
    Se toma en cuenta la Lista de Producto, el último Incremento de Producto, la capacidad del Equipo de Desarrollo y el rendimiento pasado del mismo.

El Equipo de Desarrollo son los encargados de evaluar qué elementos de la Lista de Producto son seleccionados, ya que son los únicos que estiman qué pueden lograr en el tiempo dado.
Una vez hecho esto, el Equipo Scrum elabora el objetivo del Sprint.

✔ ¿Cómo se conseguirá completar el trabajo realizado?
Una vez se elabora el objetivo del Sprint, el Equipo de Desarrollo elabora un plan para lograr el objetivo y lograr un producto terminado al final del Sprint. El dueño del Producto ayuda a clarificar elementos que sean necesarios y también puede realizar concesiones, se debe tener en cuenta que también es posible renegociar los elementos de la Lista de Producto en caso el Equipo de Desarrollo lo vea necesario, siempre junto al Dueño del Producto, como por ejemplo; si considera que tiene muy poco o mucho trabajo. El Equipo de Desarrollo puede invitar a otras personas para recibir asesoría. (pág. 10)

- **Objetivo del Sprint (Sprint Goal)**
  Es el objetivo que se traza al planificar un Sprint, es útil en muchos sentidos, entre los cuales se puede resaltar el mantener al Equipo de Desarrollo dentro de un objetivo particular y evitar que sus miembros se vayan por lados separados. Para lograr los objetivos, un Sprint, así como la Lista de Producto, tiene un conjunto de elementos por cumplir, a esto se llama Lista de Pendientes de Sprint (Sprint Backlog). (pág 11)

- **Scrums Diarios (Daily Scrums).**
  Es una reunión muy corta, un bloque de tiempo de 15 minutos que realiza el Equipo de Desarrollo cada mañana para planificar que hará durante las siguientes 24 horas. Se toma en cuenta el trabajo que se hizo después del último Scrum Diario. Se realiza a diario a la misma hora y en el mismo lugar. De acuerdo a la referencia usada, “La Guía de Scrum” (Schwaber & Sutherland, 2016), durante el mismo, cada miembro del Equipo de Desarrollo explica:

- ✔ *"¿Qué hice ayer que ayudó al Equipo de Desarrollo a lograr el Objetivo del Sprint?.*
- ✔ *¿Qué haré hoy para ayudar al Equipo de Desarrollo a lograr el Objetivo del Sprint?.*
- ✔ *¿Veo algún impedimento que evite que el Equipo de Desarrollo y yo logremos el Objetivo del Sprint?.*

- **Trabajo de Desarrollo.**

  Todo el trabajo que se realiza en un Sprint, el desarrollo, la documentación necesaria, pruebas, etc.

- **Revisión del Sprint (Sprint Review).**

  Se lleva a cabo al final de Sprint. Incluye los siguientes elementos: (pág. 12-13)

  - ✔ Asisten el Equipo Scrum y los interesados clave invitados por el Dueño del Producto.
  - ✔ El Dueño del Producto explica que elementos de la Lista de Producto se han terminado, cuáles no, y por qué.
  - ✔ El Equipo de Desarrollo habla sobre qué problemas tuvo, como los soluciono, y como aparecieron estos problemas.
  - ✔ El Equipo de Desarrollo hace una muestra del producto del incremento terminado y responde preguntas y dudas.
  - ✔ El Dueño del producto habla sobre la situación actual del proyecto y de ser necesario pronostica posibles fechas de finalización.
  - ✔ El grupo completo colabora en que se debe hacer a continuación.
  - ✔ Se revisa cómo ha cambiado el mercado externo para evaluar posibles cambios que podrían añadir más valor.
  - ✔ Revisión de cronogramas, presupuestos, etc. para la próxima entrega del siguiente Sprint.

  El resultado es una Lista de Producto revisada, mejorada y la posible Lista de Pendientes de Sprint para el siguiente.

- **Retrospectiva del Sprint (Sprint Retrospective).**
  Una vez terminado un Sprint, el Equipo Scrum puede autoanalizarse para mejorar sus procesos en los siguientes Sprint. Se lleva a cabo después de la Revisión de Sprint y antes del comienzo del siguiente Planificación de Sprint. Su propósito es: (pág. 13)
  - ✔ *"Inspeccionar cómo fue el último Sprint en cuanto a personas, relaciones, procesos y herramientas.*
  - ✔ *Identificar y ordenar los elementos más importantes que salieron bien y las posibles mejoras; y finalmente.*
  - ✔ *Crear un plan para implementar las mejoras a la forma en la que el Equipo Scrum desempeña su trabajo."*

Durante un Sprint no se realizan cambios que puedan afectar al Objetivo del Sprint (Sprint goal), los objetivos de calidad no disminuyen, y el alcance del Sprint puede renegociarse entre el Dueño del Producto y el Equipo de Desarrollo de acuerdo a la experiencia alcanzada a medida que se desarrolla el proyecto.

Un Sprint puede cancelarse antes de que el bloque de tiempo concluya, sin embargo se analiza el producto avanzado hasta ese momento, y si hay partes del producto potencialmente entregables, el Dueño del Producto puede aceptarlos. Un Sprint puede cancelarse si se dan los siguientes requisitos:

- Solo el Dueño del Producto puede cancelarlo (aunque puede ser influenciado).
- El objetivo del Sprint queda obsoleto.

Artefactos de Scrum:

Los artefactos de Scrum son similares a los que utilizan las metodologías convencionales en cuanto a función, sin embargo en cuanto a diseño y función son distintos. Los artefactos en Scrum se centran en *"representar trabajo o valor en diversas formas que son útiles para proporcionar transparencia y oportunidades para la inspección y adaptación."*(pág. 14)

Estos son:

- **Lista de Producto (Product Backlog):** Es la fuente de requisitos a llevar a cabo para el proyecto. Es una lista diseñada por el Dueño del producto; donde representa todo lo que se sería necesario para desarrollarlo, el Dueño también se encarga del orden que esta tenga y el contenido.
  Es bueno tener en cuenta los siguientes elementos al momento de realizar una Lista de Producto:
  - ✔ Cambia constantemente dependiendo de las necesidades que se tenga para mejorar el producto.
  - ✔ A medida que el proyecto avanza, esta se vuelve más larga debido a la retroalimentación proporcionada por el entorno, la experiencia ganada, la tecnología, etc.
  - ✔ El refinamiento de la Lista de Producto es un proceso continuo. El Equipo Scrum define cuando y cómo se refina, y usualmente no consume más del 10% de la capacidad del Equipo de Desarrollo, pero debe tenerse en cuenta que el Dueño del Producto puede actualizar esta lista en cualquier momento. (pág. 14)
  - ✔ El Equipo de Desarrollo es el encargado de brindar las estimaciones a los elementos de la Lista de Producto.
  - ✔ En cualquier momento es posible revisar el trabajo restante para lograr el objetivo

- **Lista de Pendientes de Sprint (Sprint Backlog):** Es el conjunto de elementos extraídos de la Lista de Producto para el Sprint a realizar, más la planificación realizara para lograr el objetivo del Sprint.

  Se debe tener en cuenta que una Lista de Pendientes de Sprint cuenta con el detalle suficiente para mostrar de manera entendible; todo el trabajo que se va a realizar durante el Sprint, además de que al igual que la Lista de Producto, es dinámica.

Cuando se necesita hacer algo que no estaba en la Lista de Pendientes inicial, el Equipo de Desarrollo lo agrega a la misma; y cuando algún elemento se considera innecesario es quitado. Esto gracias a que solo el Equipo de Desarrollo puede hacer modificaciones en la misma, por lo tanto puede mantener la lista actualizada de acuerdo a sus necesidades, mostrando la realidad del trabajo faltante y el realizado dentro del Sprint.

De manera similar a la Lista de Producto, en cualquier momento se puede estimar el tiempo restante necesario para cumplir con el objetivo del Sprint, sumando las tareas realizadas y las faltantes.

- **Incremento:** Es muy importante tener muy claro el significado de este artefacto, la Guía de Scrum lo define textualmente como:

  "El incremento es la suma de todos los elementos de la Lista de Producto completados durante un Sprint y el valor de los incrementos de todos los Sprint anteriores. El incremento debe estar en condiciones de utilizarse sin importar si el Dueño del Producto decide liberarlo o no".(pág. 16)

Producto terminado: Es definido por el Equipo de Scrum, por lo tanto, su definición va a variar constantemente, pero no del todo, ya que el objetivo del término es cumplir la función de dejar en claro cuándo un Sprint puede decir que ha cumplido su objetivo. Lógicamente a medida que los Equipos Scrum maduran y mejoran, la definición también sufre el mismo proceso, por lo que siempre debe estar bien definido en las distintas etapas, y debe ser claro para todo el equipo. (pág. 17)

2.1.13 Aplicación Móvil: Una aplicación móvil es un programa informático diseñado y programado para ejecutarse en dispositivos móviles; como teléfonos inteligentes o tabletas. Simplificando, una aplicación móvil es software, con el detalle que son desarrollados específicamente para dispositivos móviles. (Fuente propia)

2.1.14 Internet: Internet es la red de redes. Se le llama así porque interconecta a las personas alrededor del mundo a través de equipos informáticos. Para llevar a cabo esta conexión, no solo conecta a los equipos informáticos personales unos con otros, sino también a millones de servidores web; los cuales brindan información a sus usuarios por medio de alojar páginas web. Para acceder a la Internet es necesario (ademas de tener un equipo informático) tener un proveedor de este servicio. Hoy en día se ha convertido en una necesidad para poder llevar a cabo las distintas actividades diarias, es parte de la vida para gran parte de los habitantes del planeta, es usada para buscar empleo, obtener información, acceder a noticias, trabajar, enviar correos, pagar recibos, entretenimiento, estudios, salud; y la lista sigue y sigue. (Fuente propia)

2.1.15 Servidor Web: Es un software diseñado para permitir la interacción entre computadores. Implementa el protocolo HTTP para generar vistas compatibles con navegadores web, los cuales muestran de forma amigable la información que reciben. Actúa de puerta de enlace para muchos servicios, como por ejemplo el correo o FTP, ademas de encargarse de establecer conexión con distintos gestores de bases de datos. También implementa lenguajes de programación; permitiendo así la compilación en tiempo real de algoritmos generados por los programadores, para de esta manera ejecutar tareas de manera segura y haciendo uso de los recursos del equipo informático en donde se encuentran instalados. (Servidores Web)

2.1.16 Página Web: Una página web es un conjunto de líneas de código que son interpretadas por los navegadores web para mostrar información. Utilizan el lenguaje HTML como base, sin embargo, dada la gran cantidad de necesidades que se tiene hoy en día, hacen uso de lenguajes de programación para satisfacer las necesidades que se tengan. Existen muchos lenguajes de programación para desarrollar páginas web, pero para empezar es necesario aprender tres, HTML, Javascript y CSS, el segundo para poder hacer uso de los recursos del computador para ejecutar algoritmos de manera local (no haciendo uso del servidor web) o

asíncronamente, y el último para generar las hojas de estilos necesarias para dar el formato deseado a las páginas web. (Fuente propia)

2.1.17 Navegador Web: Un navegador web es un programa informático (software) que permite a sus usuarios navegar por la Internet. Recibe líneas en HTML (ver 2.1.16, Página web), y muestra en las pantallas la información en el formato deseado para poder visualizar de manera amigable lo que recibió. Es necesario para poder acceder al Internet, aunque no es imprescindible para hacer uso de la infinidad de recursos que se tiene en la red de redes. Entre los más conocidos tenemos a Mozilla Firefox, Chrome, Opera, Safari y a Internet Explorer, sin embargo hay otros menos conocidos que brindan algunas mejoras en cuanto a la experiencia de navegación y seguridad, como por ejemplo Brave. (Fuente propia basado en Qué es una navegador Web. s.f.)

2.1.18 IDE (Integrated Development Environment): En español significa "Entorno de Desarrollo Integrado" o "Ambiente de Desarrollo Integrado". Como su nombre lo indica, es un programa informático que brinda las herramientas necesarias para programar en uno o más lenguajes de programación. Constan de un editor de códigos, un compilador, un depurador y un constructor de interfaces gráficas. (Entorno de Desarrollo Integrado. 2011)

Los IDEs proveen ambientes cómodos para el desarrollo, y entre los más comunes tenemos:

- Android Studio.
- Netbeans.
- Aptana.
- Visual Studio.
- Zend Studio.
- Eclipse.

2.1.19 Android Studio: Es el IDE (Integrated Development Environment, en español, Ambiente Integrado de Desarrollo) oficial de Android, un software integrado que brinda todas las herramientas para desarrollar aplicaciones tanto para cualquier dispositivo que cuente con el sistema operativo Android. Posee grandes cualidades como por ejemplo implementar un simulador donde se puede programar un dispositivo móvil o una tableta y ejecutar las líneas de código que se van programando. Integra entre otras cosas, el diseño de interfaces basados en XML, y Java para el resto de procesos, aunque poco a poco está integrando Kotlin, el nuevo lenguaje para programar en Android. (Android Studio)

2.1.20 Netbeans: Es un IDE conocido a nivel mundial para trabajar con Java, sin embargo no se limita a solo Java, también es excelente para trabajar páginas web basadas en PHP, por lo que integra HTML, Javascript y CSS de manera muy buena. Posee una excelente gestión de proyectos y reconoce errores además de enlazarse con las distintas carpetas del proyecto para facilitar la escritura de funciones y clases. En el presente proyecto es el IDE usado para programar todo el sistema backend. (Netbeans)

2.1.21 Lenguaje de Programación: El diccionario de la Real Academia de la Lengua Española (RAE), define al lenguaje (cuando se refiere a Informática) como: *"Conjunto de signos y reglas que permite la comunicación con la computadora"*, y al "Lenguaje Máquina" como el *"Conjunto de instrucciones codificadas que una computadora interpreta y ejecuta directamente".* (Lenguaje).

Extendiendo un poco la definición, un Lenguaje de Programación es un conjunto de reglas y palabras clave que en conjunto; ordenan a un computador a realizar tareas en un orden establecido. Existen muchos lenguajes de programación, algunos mas conocidos que otros, de bajo (como el lenguaje ensamblador) o alto nivel (como Java), y con propósitos muy particulares (como Matlab o Swift).

2.1.22 Paradigma de Programación: Antes de adentrarse en el termino en toda su complejidad, es bueno dejar en claro qué es un paradigma. La RAE define a paradigma como: *"Teoría o conjunto de teorías cuyo núcleo central se acepta sin cuestionar y que suministra la base y modelo para resolver problemas y avanzar en el conocimiento"* (Paradigma. s.f).

Adentrándonos más en una definición más orientada a la Ingeniería de Sistemas, es una manera particular de estructurar las líneas de codigo al momento de programar, de manera que se mantiene una estructura y organización para llevar a cabo los procesos. Están relacionados estrechamente a estilos de programar. Existen varios paradigmas, entre los que podemos destacar a la programación Estructurada, Orientada a Objetos, Orientada a eventos, Asociados a la concurrencia, etc. (Paradigmas de Programación, 2011)

2.1.23 POO (Programación Orientada a Objetos): La POO es un paradigma de programación que hace uso de los objetos, clases y las interacciones de en el diseño de un sistema. Está compuesto por:

- Clase: Es un modelo usado para, en base al mismo, crear objetos que comparten similares características.
- Objeto: Es una entidad provista de ciertas características propias y comunes (atributos), además de métodos.
- Métodos: Son algoritmos asociados a un objeto, mediante los cuales se puede saber qué acciones realiza ese objeto.
- Eventos: Son sucesos que ocurren dadas ciertas reglas, y que envían mensajes indicando lo que ha ocurrido.
- Propiedades y Atributos: Son los datos asociados a un objeto; en otras palabras sus características.

Las características de la POO son:

- Abstracción: Es la capacidad de extraer las características y métodos que tiene un objeto.

- Encapsulamiento: Es la capacidad que tiene la POO para agrupar los atributos de secciones dependiendo de su importancia y comportamiento.
- Modularidad: Es la capacidad de dividir una aplicación en partes mas pequeñas que interactúa entre si, permitiendo entre otras cosas, que el mantenimiento del sistema sea mas sencillo y que la escalabilidad del mismo sea mayor.
- Ocultación (Aislamiento): Es la capacidad de proteger los atributos y métodos de un objeto, indicando de manera sencilla quienes pueden acceder a los mismos y quiénes no.
- Polimorfismo: Es la capacidad de contar con métodos que se adecúen a las necesidades del momento sin necesidad de usar distintos nombres, dependiendo de los parámetros recibidos para cumplir similares o distintas funciones.
- Herencia: Es la relación que existe entre una o más clases, donde una clase “hija” hereda los métodos y atributos de una clase “padre”, además de ciertos privilegios.
- Recolección de Basura: Es una implementación dentro de los lenguajes Orientados a objetos que permite destruir objetos y datos ubicados en memoria cuando estos ya no se necesito, liberando espacio de memoria. (Bahit E. s.f. págs. 11-12)

2.1.24 PHP: A modo de dejar en claro rápidamente qué es PHP, partimos con una breve explicación para entrar en contexto rápidamente. Es un lenguaje de programación de lado servidor usado nivel mundial tanto tanto en el ambiente académico como en el ambiente empresarial. Entre sus características se puede mencionar que es multiplataforma y que es casi orientado a objetos. Existe una gran cantidad de librerías para PHP, y la madurez que ha logrado a través de los años ha hecho de este lenguaje uno de los preferidos a nivel mundial. (Coronel E. 2010. Pág 11).

Ampliando un poco mas sobre este lenguaje, PHP es el acrónimo recursivo de “PHP Hypertext Preprocessor”, que en español significa “PHP

Preprocesador de hipertexto" (¿Qué es PHP?. s.f.). Sin embargo esto no es del todo cierto, para entender mejor cómo llega a ese nombre es necesario ir más atrás. Fué creado el año 1994 por Rasmus Ledorf, este personaje creó unos scripts para rastrear las visitas que recibía su curriculum online, y llamó a ese conjunto de scripts "Personal Home Page Tools" (en español; "herramientas de la página de inicio personal"), posteriormente en el 1995; Rasmus publicó el código fuente de PHP Tools, logrando así que otros programadores se interesen en el mismo y empiecen a usarlo.

El nombre PHP cambió en Septiembre del 1995 a FI (Forms Interpreter, que en español significa; Interpretador de Formularios), y con las mejoras que Ledorf realizó, se volvió mas potente y ya incluía algunas herramientas que son útiles hasta el día de hoy, sin embargo el nombre no duró mucho, y en Octubre empezó a llamarse, resumidamente, "Personal Home Page Construction Kit" (en español; Kit de construcción de páginas de inicio personales). Las modificaciones siguieron y posteriormente salieron versiones de PHP, como por ejemplo PHP 3, que fué la primera versión que más se parece al actual PHP. (Historia de PHP. s. f.) Posteriormente salieron Php 4, 5, 6 y actualmente existe la versión 7 que poco a poco se vuelve más y más popular.

2.1.25 Java: Es un lenguaje de programación ampliamente usado en el mundo de hoy, con un crecimiento importante y que ha ido evolucionando muy satisfactoriamente a lo largo de los años. Originalmente fué creado por Sun Microsystems; pero actualmente es propiedad de Oracle. Es un lenguaje orientado a objetos de propósito general, y aunque fue conocido como como un lenguaje de programación de applets, es usado también para construir cualquier tipo de proyectos, y gracias al auge de Android, ha crecido en popularidad. (Nolasco J. 2013. pág. 15)

Java cuenta con una gran comunidad de desarrolladores a nivel mundial, alrededor de 9 millones, además el 89% de computadoras en Estados Unidos ejecutan Java. Para hacerlo aún más atractivo; 3 mil millones de

móviles ejecutan Java y cerca de 15 millones de dispositivos de televisión también. Para entender estos números hay que tener en cuenta que Java se ejecuta en prácticamente todos los escenarios, y en la web se ha posicionado bastante bien, y si agregamos a esto que existe una serie de capacitaciones brindadas por Oracle mismo mediante el “Oracle Academy”, donde uno puede certificarse y seguir cursos, se entiende por qué ha crecido y sigue siendo tan importante hasta el día de hoy. (Conozca más sobre la tecnología Java. s.f)

2.1.26 XML (Extensible Markup Language): En español significa “Lenguaje de Etiquetado Extensible”, es similar al HTML, pero su función principal es describir mas no mostrar datos como hace HTML. Es capaz de agrupar información dentro de etiquetas de manera que esta sea fácil de leer y entender, además de brindar un formato ordenado y fácil de interpretar por los distintos lenguajes de programación. (Guía breve de Tecnologías XML. s.f.)

En la figura 6 se puede apreciar un ejemplo del mismo:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<libro>
  <titulo></titulo>
  <capitulo>
    <titulo></titulo>
    <seccion>
      <titulo></titulo>
    </seccion>
  </capitulo>
</libro>
```

Figura 6. Captura de código de ejemplo en XML.

Como se puede apreciar en la figura Nro. 6, un libro cuenta con ciertos datos, entre ellos el título y sus capítulos, este último a su vez cuenta también con un título y una sección, y cada sección del mismo cuenta con otro titulo. Esa es la manera de describir un libro haciendo uso de XML.

2.1.27 Programación en Android: Hasta hace pocos años el IDE ideal para la programación en Android era Eclipse, pero con el paso del tiempo esto ha cambiado, Android Studio ha ido ganando terreno, ha mejorado versión a versión y actualmente es el IDE oficial para el desarrollo en Android, por lo que en el presente proyecto se programó usando Android Studio (ver 2.1.19)

Al programar en Android hay algunos puntos que se tienen que tener en claro, entre ellos se encuentran:

2.1.27.1 Estructura de un proyecto en Android Studio:

La estuctura de un proyecto en Android Studio varía dependiendo de la vista que se este usando, en esta investigación se utilizo la vista "Android Studio", en la cual la estructura se ve tal cual se representa en la figura 7.

Existe cierto orden predeterminado para los distintos archivos, si bien algunos archivos no necesariamente deben llamarse o estar en esas ubicaciones, a modo de estandarizar el desarrollo y mantener un orden; es que se ubican y llaman de esa manera.

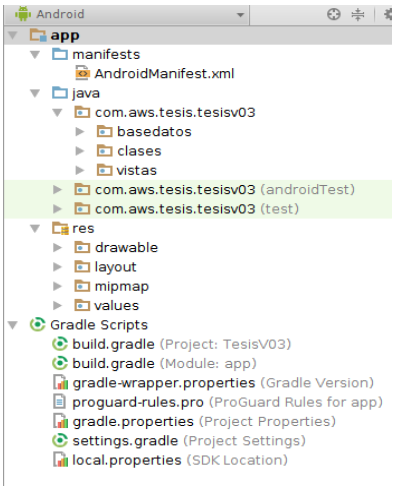

Figura 7. Estructura de un proyecto en Android Studio. Basado en el IDE Android Studio

2.1.27.2 Manifests: La carpeta “manifest” contiene un archivo llamado AndroidManifest.xml, en el cual, se declararon todos los “layouts” usados, permisos requeridos, configuración para compatibilidad de 32 y 64 bits, orientación de los layouts y el ícono de launcher de la aplicación.

2.1.27.3 Java: Dentro de la carpeta “java”, propiamente dentro de la primera carpeta de la misma, es donde se alojan las distintas clases que utilizan para programar, allí está la mayor parte del código, y se ordena de acuerdo a las necesidades del programador.

2.1.27.4 Res: Dentro de la carpeta “res” se aloja los distintos recursos usados, ya sean imágenes, vistas (layouts), el ícono de la aplicación, y todos los valores que se vayan a usar, tanto para estilos como para textos, dentro de “values” también se alojan los archivos de texto para los distintos idiomas de la aplicación.

2.1.27.5 Layouts: Para en desarrollo en Android Studio, los layouts son las interfaces. Estas se desarrollan en XML de dos maneras, usando la interfaz gráfica que provee el IDE, o mediante líneas de código. Ambos tienen sus potencialidades, sin embargo el uso de uno u otro depende del programador o las necesidades propias del proyecto. En los layouts se diseña las interfaces, y esto incluye todos los elementos que podría necesitar una interfaz, algunos ejemplos son: botones, etiquetas (labels), fragmentos y entradas de texto.

2.1.27.6 Drawable: Es la carpeta donde se almacenan todos los recursos gráficos, incluyendo el ícono del launcher. En el desarrollo para Android, las imágenes se clasifican de una manera particular, esto es debido a que los distintos dispositivos existente no tienen la misma resolución ni la misma densidad de píxeles; por lo tanto no se recomienda usar una sola imagen para cierta necesidad, sino hasta seis imágenes. Esto

podemos apreciar mejor en la figura 8 (“ldpi” está casi obsoleto para teléfonos inteligentes):

ldpi (low) ~120dpi

mdpi (medium) ~160dpi

hdpi (high) ~240dpi

xhdpi (extra-high) ~320dpi

xxhdpi (extra-extra-high) ~480dpi

xxxhdpi (extra-extra-extra-high) ~640dpi

Figura 8. Densidades generalizadas. Basado en el artículo de la web oficial de Android; Compatibilidad con diferentes pantallas. Fuente: Compatibilidad con diferentes pantallas. 2018.

De acuerdo a los datos mostrados en la figura 8, se debe tener en cuenta cuándo una imagen se usará dependiendo del “dpi” del dispositivo, por lo tanto al momento de generar o descargar imágenes, estas se almacenan dependiendo de a que “dpi” van orientadas. En la siguiente imagen se ve un ejemplo de cómo se organizan. Nótese en la figura 9 la ruta en la parte superior para entender mejor donde estas carpetas de imágenes se encuentran.

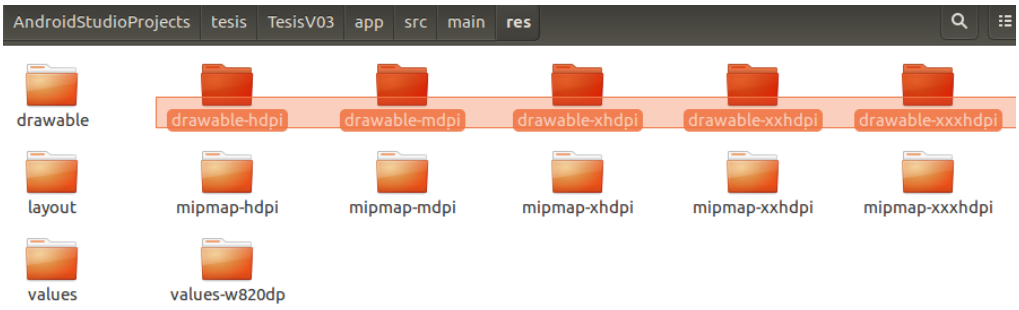

Figura 9. Representación de cómo se almacenan las imágenes para Android Studio. Fuente propia.

2.1.27.7 Values: Dentro de la carpeta “values” se recomienda almacenar valores que se utilizan en la aplicación en formato XML. En este proyecto dentro del archivo “colors.xlm” se especifican los valores de

los colores utilizados en formato hexadecimal, dentro del archivo “strings.xml” se especifican todos los valores de textos no dinámicos (como labels) usados en la aplicación, y en el archivo “styles.xml” se especifican estilos de la aplicación.

Se debe tener en consideración que se puede crear archivos de valores de acuerdo a las necesidades del desarrollador, y se recomienda en todo caso mantener un orden coherente y no saturar de archivos innecesarios esta carpeta para tener un fácil mantenimiento.

2.1.27.8 Mipmap: Dentro de esta carpeta se almacenan los launchers de la aplicación en sus distintas densidades de pixeles.

2.1.27.9 Graddle: Finalmente, todos los archivos dentro de “Graddle scripts” son generados y mantenidos automáticamente por Android Studio y se recomienda no modificar. Allí se encuentra la información sobre la versión mínima de Android para la cual se programa, la versión objetivo (datos que se eligen al momento de crear un nuevo proyecto), dependencias y también configuración para compatibilidad de 32 y 64 bits, siendo estos dos últimos modificados para poder realizar el proyecto.

2.1.28 HTML (HyperText Markup Language): En español significa “Lenguaje de Marcas de Hypertexto”. Si bien tiene la palabra “Lenguaje” dentro de su nombre, hay que tener en claro que no es un lenguaje de programación, sino un lenguaje de marcado. HTML ordena una página web dentro de nodos que a medida que se van generando, van ordenándose (valga la redundancia) mediante el orden que se van ingresando y el uso de hojas de estilo (css), mostrando o solicitando información mediante texto, imágenes o formularios; que se ingresan de manera manual o automática. (Qué es HTML. s.f.)

Las etiquetas que utiliza varían dependiendo de las necesidades del usuario y el estándar existente, contiene ciertas reglas que deben seguirse para el correcto funcionamiento, entre ellas la más importante es la manera de abrir y cerrar etiquetas. Para abrir una etiqueta, esta debe estar entre los símbolos '<' y '>', y para cerrarla, debe estar entre los símbolos '</' y '>', por ejemplo:

<abriendo etiqueta> </cerrando etiqueta>

2.1.29 CSS (Cascading Style Sheets): En español significa "Hojas de estilo en Cascada', y como su nombre lo indica, son los encargados de brindar los estilos de una página en HTML. CSS es un lenguaje de estilo, y es ampliamente soportado por los navegadores existentes. (Lección 1. ¿Qué es CSS?. s.f)

Es capaz de dar el formato que uno desee a una página web, y brinda estilos a todas las etiquetas disponibles en HTML. También da formato a las etiquetas que cuentan específicamente con un "ID" o una "clase", mediante los símbolos '#' y '.' respectivamente.

Una de las potencialidades más grandes de CSS es que, si se usa correctamente, ahorra muchísimas líneas de código gracias a su capacidad de agrupar etiquetas e indicar reglas para todas ellas en un solo bloque de código, por ejemplo: Si se quiere que las etiquetas "p" y "span" tengan el mismo tamaño y color de letra, basta con utilizar el siguiente código:

```
p, span{
   color: #f1f1f1;
   text-size: 15px;
}
```

De esa manera no se debe crear las mismas reglas para "p", y para "span".

2.1.30 JavaScript: Es un lenguaje de programación usado especialmente para la creación de páginas web dinámicas. Es ampliamente usado por los desarrolladores ya que este se ejecuta en la máquina del usuario mas no en el servidor, por lo tanto no consume recursos fuera del dispositivo del usuario (aunque contiene funciones que pueden consumir recursos de servidor). Tiene tres partes: (Zakas, N. 2012. pág. 2)

- ECMAScript: Tras la creación de JavaScript, el lenguaje fue estandarizado por ECMA (European Computer Manufacturers Association, en español: Asociación Europea de Fabricantes de Computadoras), cuyo propósito fue estandarizar la semántica, sintaxis de un lenguaje de de propósito general, que funcione en varias plataformas y que no tenga propietarios. Básicamente describe: (pág. 3)
  - Sintaxis.
  - Tipos.
  - Declaraciones.
  - Palabras reservadas.
  - Operadores.
  - Objetos.
  - Funciones.
  - Propiedades.

- DOM (Document Object Model): Es una API para XML, que básicamente mapea todo el contenido de una página web en nodos. De esta manera se puede ubicar cualquier línea de código o elemento que contiene una página web. También brinda soporte para el mouse y todos los eventos que pueden ocurrir en una web, y describe interfaces que soportan CSS. (págs. 6-8)

- BOM (Browser Object Model): Permite la manipulación de la ventana del navegador. Entre las mayores ventajas que ofrece se tiene: (pág. 9-10)
  - Capacidad de crear pop ups.
  - Capacidad de mover, redimensionar y cerrar ventanas del navegador.

- Soporte para cookies.

2.1.31 JSON (JavaScript Object Notationt): En español significa “Notación de objetos de JavaScript”, y es actualmente un estándar para mostrar datos. Su sintaxis es la siguiente: (Zakas, N. 2012. pág. 691-694)

- **Valores simples:** Parar mostrar valores simples como cadenas de texto (las cadenas de texto van entre comillas), números (ejm: 5), booleanos (ejm: true) y null (representado por un vacío).

- **Objetos:** Los objetos se representan tal cual se ve en la figura 10.

{"estado":1,"data":[{"id":"1","nombreEnUnidad":"Servicio Rapido","nombreLegal":"Servicio Rapido S.A.","ruc":"20278567568","TRutas_id":"1"},{"id":"2","nombreEnUnidad":"Liebre","nombreLegal":"Liebre SRL","ruc":"20564086062","TRutas_id":"2"}]}

Figura 10. Captura de resultado de JSON aplicado en objetos.

La figura 10 muestra el formato en que el Servicio Web brinda información de las empresas ingresadas al sistema, “estado” indica, en caso de ser ‘1’, que la respuesta del servicio web fue correcta, y posteriormente agrupa dentro de corchetes los datos de cada empresa ingresada en el sistema y la ruta que se le asignó.

- **Arreglos:** Los arreglos se muestran de manera similar a los objetos

2.1.32 Servicio Web: En inglés llamados “Web Services”. Son un conjunto de protocolos usados para intercambiar información entre aplicaciones, por ejemplo, si alguien quisiera brindar información sobre el clima en una ciudad, tendría que tener un formato mediante el cual las personas que quieran conocer esta información (consumir) puedan entender el clima que habrá, por lo que una aplicación que existe por ejemplo en el internet, puede

comunicarse con una aplicación dentro de un teléfono con Android de manera ordenada y sencilla.

La manera que un Servicio Web muestra los datos, es a través de XML, y hoy en día el uso de JSON para mostrar los datos de un servicio web se ha vuelto una práctica común entre los desarrolladores que necesitan consumir un servicio web para Android.

2.1.33 Sistema Gestor de Bases de Datos (SGBD): Un SGBD (DBMS en inglés), o también conocido como "Sistema de Gestión de Bases de Datos" es un programa que permite al usuario final; comunicarse de manera segura y simple, haciendo uso de algún lenguaje, con los datos que almacena. En pocas palabras, insertar, borrar, actualizar o consultar información. Los SGBD gestionan tres cosas importantes: los datos, el motor de base de datos y el esquema de la base de datos.

Entre las ventajas que se tiene al usar un SGBD tenemos:

- Seguridad.
- Mecanismo de bloqueo para evitar concurrencia.
- Un controlador que eficientemente balancee las necesidades distintas de los distintos programas que hacen uso de la misma.
- Habilidad de recuperarse de errores, etc

Por otro lado, el uso de un SGBD permite al usuario centralizar su información de manera más segura y mucho más eficiente en cuanto al acceso y manejo en general de la misma, ya que estos sistemas están creados para manejar eficientemente estos procesos. (Database Management Systems. s.f.)

2.1.34 Bases de Datos Relacionales: Las bases de datos relacionales son aquellas que se basan en conjuntos de tablas y se manipulan de acuerdo al modelo relacional (Bases de datos relacionales). En la presente tesis se

utiliza MySql como sistema gestor de base de datos (véase 2.1.36), el cual también utiliza bases de datos relacionales.

2.1.35 SQL: Es el lenguaje estándar para acceder y manipular bases de datos, es un estándar de la American National Standars Institute (ANSI, en español; Instituto Americano Nacional de Estándares), y es la base para consultar y manipular gestores relacionales de bases de datos. SQL, al ser un lenguaje, reúne una serie de reglas e instrucciones para poder ejecutar distintas órdenes dentro de gestores de bases de datos, y se convirtió en el estándar de grandes gestores de bases de datos como; MySql, MS SQL Server, Oracle, Microsoft Access y SQLite. (Introducción a SQL. s.f.)

2.1.36 MySql: Es un SGBD distribuido bajo la licencia GPL. Actualmente es propiedad de Oracle, y de acuerdo a su página oficial, es la base de datos de código abierto más popular, además de que es usada por empresas de renombre como Facebook, Twitter, Youtube, Yahoo y muchas más. (About My Sql)

MySql es un SGBD Relacional (SGBDR), sin embargo de acuerdo a ciertas definiciones, MySql carece de ciertas características que lo harían completamente relacional, pero en esencia; es relacional. Cuando hablamos de “Relacional”; nos referimos a que es capaz de crear relaciones de dependencia entre las tablas que se generan, dependencias que aseguran que los datos ingresados sean tratados de manera correcta y que no ocurran errores lógicos. (Bell, C. 2012. pág 27)

Muchas de las aplicaciones creadas que utilizan SGBDR, necesitan desarrollar sus propios algoritmos para poder comunicarse con MySql, utilizando una serie de protocolos llamados “Conectores de base de datos”, los cuales usualmente se basan en el modelo ODBC (Open Database Connectivity, en español; Conectividad de bases de datos abiertas). En el caso particular de PHP y Java, existen conectores creados tanto por MySql

como por la comunidad, en caso de Java se usa JDBC, y en caso de PHP se utilizan diversos controladores como mysqli o PDO_MySql. (pág. 28-29).

El código fuente de MySql está creado en los lenguajes de programación C y C++, y si bien no esta orientada a objetos, esta creada de manera que sea altamente eficiente y confiable. (pág 38) Vale la pena destacar que también implementa InnoDB (es un motor de almacenamiento comprado por Oracle), y gracias a esto, es capaz de dar soporte a las cláves foráneas entre otras cosas.

2.1.37 SQLite: Es un SGBD con la característica particular de no ser tan complejo ni pesado como otros que existen en el mercado, y que no se ejecuta de manera paralela al programa que lo utiliza, sino que coexiste con la aplicación que hace uso del mismo y a su vez sirve de anfitrión. Una gran ventaja de tener a SGBD dentro de un programa, es que no se necesita de configuración o administradores de redes. (Grant A. & Owens M. 2010. pág. 1)

SQLite es software libre, no tiene ningún tipo de licencia ni de copyright, y todos quienes contribuyen con SQLite deben rechazar cualquier interés en copyrights, por lo que cada persona que lo utilice puede modificarlo a su antojo y lo mismo con su utilización, con ninguna restricción. (pág. 9)

SQLite tiene la enorme cualidad de ser muy ligero, y no por eso es decuidado; ya que está muy bien escrito, y de acuerdo a su página web (www.sqlite.com); es la base de datos más desplegada, con más aplicaciones que la utilizan que lo que se puede contar, incluyendo proyectos grandes. Por otro lado; de manera similar a otros SGBD más grandes, SQLite maneja las transacciones de manera adecuada siguiendo lo siguientes parámetros: (About SQLite)

- Atomicidad: Las transacciones de ejecutan de principio a fin de manera indivisible, todo o nada.

- Consistencia: Toda modificación hecha en la base de datos cumple con las reglas que se establecen en la misma.
- Aislamiento: Significa que cada transacción que se realiza se ejecuta una después de otra, evitando así la concurrencia.
- Durabilidad: Significa que una vez que una transacción se realiza, esta se almacena permanentemente. (Grant A. & Owens M. 2010. págs. 11-13)

SQLite no soporta todos los tipos de datos que MySql. La tabla 3 indica cómo se pueden definir los mismos en SQLite.

Tabla 3

Conversion de datos entre MySql y SQLite

Tipo	Afinidad
INT INTEGER TINYINT SMALLINT MEDIUMINT BIGINT	INTEGER
CHARACTER(20) VARCHAR(255) TEXT CLOB	TEXT
BLOB	BLOB
REAL DOUBLE FLOAT	REAL
NUMERIC DECIMAL(10,5) BOOLEAN DATE DATETIME	NUMERIC

Nota: Basado en “Datatypes In SQLite Version 3”

2.1.38 Workbench: Es una herramienta visual unificada para arquitectos de bases de datos, desarrolladores y administradores de bases de datos. MySql Workbench provee modelado, desarrollo SQL y herramientas de

administración exhaustiva para configuración de servidores, administración de usuarios, backups y mucho más. Se encuentra disponible para Windows, Linux y Mac OS, y actualmente es propiedad de Oracle. (MySql Workbench)

2.1.39 StarUML: Es un software muy sofisticado para generar diagramas UML, cuenta con más de 5 millones de descargas y puede ser usado tanto en Windows, Linux y Mac. Este programa cuenta con una característica muy particular, si bien no es gratuito, permite ser usado sin limitaciones y sin tener la obligación de pagar. (StarUML)

2.1.40 Gimp: Es un editor de imágenes gratuito disponible para Linux, Windows y Mac. Gimp es software libre, por lo que es posible cambiar su código fuente y distribuirlo. La comunidad de usuarios de Gimp crece día a día, y gracias a ello existen muchos plugins para el mismo. Su nombre es el acrónimo de GNU Image Manipulation Program (Programa de Manipulación de Imágenes de GNU) (Gimp)

2.1.41 Libre Office: Es un software creado para editar documentos al estilo de Microsoft Office, en otras palabras un paquete ofimático, con la enorme de diferencia de ser gratuito y de código abierto (es software libre), incluye un editor de textos llamado Writer, una hoja de cálculo (Calc), software para presentaciones (Impress), entre otros. En el presente proyecto, es el paquete de ofimática usado. (Libre Office)

2.2 Antecedentes de la Investigación

Los antecedentes encontrados para el presente tesis destacan 3 aspectos muy importantes, primero; la uztilización del PUDS, segundo; la utilización de SCRUM, y tercero; el desarrollo para Android. Se tomaron en cuenta tanto tesis nacionales como internacionales, además de aplicativos existentes con funciones similares al del presente proyecto.

La tesis del bachiller Francesco Galiano Abanto, de la Universidad Andina del Cusco (Perú), titulada “Scrum como metodología de desarrollo de software para el control

de caja en el restaurante Il Giardino del Cusco" es uno de los antecedentes tomados especialmente por dos motivos, la sencillez con que explica los distintos procesos de Scrum y su aplicación; y la sencillez con que genera los distintos diagramas que necesita documentar el producto. La metodología que se usa para la realización de la mencionada tesis es Aplicada, con un diseño transeccional descriptivo.

La tesis del bachiller Jorge Fabrisio Cornejo Aramayo, de la Pontificia Universidad Católica del Perú, titulada "Análisis, diseño e implementación de una aplicación para administrar y consultar avisos clasificados para tabletas Android", es un claro ejemplo de desarrollo para Android donde además de tener una base de datos en línea, se tiene una base de datos en el teléfono, generando sincronía en ambos. La metodología de desarrollo de software que se usa en la mencionada tesis es RUP, sin embargo también basa su trabajo en el PMBOK desarrollado por PMI.

La tesis de los bachilleres Mariela A. Afonso R. y Jesus E. Segnini R, titulada "Desarrollo de un sistema automatizado bajo entorno web para el control de la programación académica en la Universidad del Oriente Núcleo de Anzoátegui", de la Universidad de Oriente (Venezuela), desarrollaron una tesis muy extensa y detallada usando el PUDS.

Por otro lado, existen varios programas similares al desarrollado en el presente proyecto, estos contribuyen a la movilización de las personas en el mundo; e integran distintas tecnologías y herramientas para cumplir su función. Los que tomamos como referencia en este proyecto son:

Moovit: Es una aplicación bastante conocida a nivel mundial, de acuerdo a su página oficial es la aplicación número uno en transporte público en todo el mundo con más de 55 millones de usuarios. Si bien en el Perú es bastante desconocido debido a que por el momento sólo integra el sistema de buses de Lima, en otros países ha sido un éxito. Se la puede encontrar en la Play Store con el nombre de: "Moovit: info de bus y tren", y registra en esta plataforma más de diez millones de descargas, además cuenta con una aplicación para IOs y también una aplicación web. Moovit muestra la ruta a seguir desde un punto elegido a otro destino, además de los buses y trenes que uno debe tomar y el lugar dónde debe bajar. En las

ciudades donde está disponible, muestra los horarios exactos de llegadas y salidas, por lo que puede detallar información bastante precisa en cuanto a horarios. Se encuentra actualmente en muchos países del mundo, como Estados Unidos, Reino Unido, España, Italia, Francia, Polonia, Suecia, Finlandia, Países Bajos, Chile, Colombia, México y Perú. (Moovit)

CAPÍTULO 3 – METODOLOGÍA

A la fecha; es complicado encontrar bibliografía que indique claramente el tipo, diseño, enfoque y método de una investigación cuando se habla de desarrollo de software. Los ingenieros de sistemas, en el rubro del desarrollo de software, buscan dar soluciones a problemas mediante el desarrollo y uso de software, para ello existen metodologías y esquemas de trabajo, mientras tanto, los autores más conocidos de libros sobre metodología de investigación no enfocan sus trabajos en el software, sino mas bien en el resto de ramas que ejercen la investigación, como las ciencias de la salud o las ciencias sociales. Por lo tanto, al hablar de desarrollo de software, no se encuentra un autor que hable sobre tipo, diseño, enfoque y metodología de investigación a la vez, lo que nos obliga a consultar las obras de más de un autor.

3.1 Metodología de la Investigación

3.1.1 Tipo de Investigación

La presente tesis responde al tipo de investigación Aplicada. Esto es debido a que busca el desarrollo de software aplicando los conocimientos obtenidos en la universidad; y en la experiencia ganada en los años posteriores para así solucionar el problema de movilización que encuentran los visitantes y habitantes de la ciudad del Cusco. (Tam, Vera y Oliveros. 2008. pág.145-154)

3.1.2 Diseño de la Investigación

Antes de explicar el diseño de la investigación, se recomienda leer el punto 3.1.4. Habiendo dado la recomendación, se puede afirmar que el diseño de la presente investigación es, de acuerdo a Hernández, Fernández y Baptista (2010), de investigación-acción.

La principal finalidad del diseño de investigación-acción es la resolución de problemas cotidianos e inmediatos (pág. 509), lo cual es apoyado por el hecho que este tipo de diseño es democrático y equitativo; esto quiere decir que la población en general tiene voz y se valora la contribución de las

distintas personas (pág. 510). Además, las 3 fases de la investigación-acción son observar, pensar y actuar, las cuales se dan de manera cíclica hasta lograr el resultado esperado (pág. 511).

3.1.3 Alcance de la Investigación

El alcance de la presente tesis son los 8 distritos de la provincia del Cusco, estos son: Cusco, Ccorca, Poroy, San Jerónimo, San Sebastián, Santiago y Wanchaq. Se busca que la aplicación cubra distintas rutas que integren estos 8 distritos, así como la mayor cantidad de paraderos de los que se pueda recoger las coordenadas distribuidos entre los 8 distritos; que además sean parte de las rutas ingresadas.

3.1.4 Enfoque de la Investigación

El enfoque de la presente investigación es Cualitativa, esto debido a que al crear esta App; se busca satisfacer alguna necesidad de los clientes, y en este caso en particular; es la falta de información sobre rutas y cómo movilizarse haciendo uso del transporte público en la provincia del Cusco.

En el enfoque cualitativo se debe tener una relación cercana con el fenómeno a estudiar, que en este caso es la satisfacción del cliente, por lo tanto, la relación 'desarrollador-dueño del producto' es cercana y existe prácticamente durante todo el proceso de desarrollo. También es muy abierta a las modificaciones, ya que la realidad es modificada de acuerdo a la recolección de datos, por lo tanto el software se puede modificar dependiendo de las necesidades del cliente, adecuándose perfectamente a la capacidad de adaptabilidad de Scrum. (pág. 11-14)

3.1.5 Método de la Investigación

El método es Experimental pre-experimental. Esto quiere decir de acuerdo a Tam, Vera y Oliveros (2008. pág. 149-150): *"En este método los tratamientos de la variable independiente han sido manipulados por el investigador -X- por lo que se tiene el mayor control y evidencia de la causa-*

efecto.”, además, al no existir un grupo de control, se realiza una post-prueba y se podría realizar una pre-prueba.

3.2 Población y Muestra

En el enfoque cualitativo las muestras se dividen en varios tipos, en este caso en particular se utiliza las “Muestras mas bien orientadas hacia la investigación cualitativa”, dentro de las cuales la que se eligió fue el “Muestreo por conveniencia”, el cual apunta a individuos a los cuales tenemos acceso. (Hernández, et al. 2010 pág. 401)

Lógicamente para los fines de esta investigación, no se puede levantar información de personas que no cuenten al menos con un teléfono inteligente con sistema operativo Android, por lo cual se establecen los siguientes criterios:

- La persona tiene un smartphone con android.
- La persona cuenta con plan de datos.
- La persona hace uso del servicio de transporte público interurbano dentro de los 8 distritos de la ciudad del Cusco.

La cantidad de personas que se tomó como muestra es de 20.

3.3 Instrumentos

Se utilizaron dos instrumentos para recolectar información, historias de usuario y encuestas. Las historias de usuario fueron la base de los requerimientos (similar a los casos de uso en UML), mientras que las encuestas fueron la manera de medir la satisfacción del cliente siguiendo el enfoque cualitativo.

3.4 Recolección y Análisis de Datos

El proceso de recolección de datos se realizó en dos etapas.

- Las historias de usuario fueron obtenidas en etapas tempranas de la tesis, gracias a ellas se obtuvieron los requerimientos mas importantes a nivel “usuario” y a nivel “dueño del producto”.

- En la segunda etapa se recolectaron las impresiones de los usuarios respecto al diseño y funcionalidad del aplicativo para Android, ello para medir la satisfacción de los mismos.

También se recolectó datos provenientes de la asesora de tesis y personas a las que se fue mostrando la aplicación para Android a medida que se iba desarrollando, lógicamente estos datos fueron tomados en cuenta y muchos de ellos han sido implementados a lo largo de los sprints, sin embargo no se detallarán en esta sección, ya que si bien fueron importantes, no miden directamente la opinión de los usuarios, en otras palabras, no es igual pedir una opinión sobre un aplicativo en camino que sobre uno ya terminado o en las etapas finales.

3.4.1 Procedimiento de Recolección de Datos

3.4.1.1 Historias de Usuario:

El procedimiento para la obtención de historias de usuario fue mediante los objetivos previamente indicados más algunas ideas que pasaron el filtro y al fueron agregadas a la tesis. Las historias de usuario se detallan en el punto 4.1.3.

3.4.1.2 Encuestas:

Para recolectar datos mediante las encuestas se creó un cuestionario al momento que la aplicación estuvo lista para ser probada por usuarios. La encuesta realizada se puede ver en el anexo 1.

3.4.2 Análisis de Datos

En cuanto a las historias de usuario, al análisis de estos datos se refleja en el desarrollo de la tesis (Capítulo 4).

En cuanto a las encuestas, el análisis de datos se puede apreciar en el anexo 2.

CAPÍTULO 4 – DESARROLLO DEL SISTEMA

El desarrollo de la presente tesis depende de dos importantes conjuntos de requisitos, los del “backend” y los de “frontend”. El backend es un sistema web donde se realiza todo lo relacionado al mantenimiento de la base de datos online, la cual es encuentra alojada en un servidor web. El frontend es la aplicación para Android donde los usuarios hacen uso de la Aplicación Móvil para Consulta de Rutas del Transporte Público de la Ciudad del Cusco. Sin el backend; el mantenimiento del sistema no sería sencillo, por ejemplo; ingresar, modificar o eliminar coordenadas, una tras otra, sería complicado y repetitivo, además; el backend ayuda muchísimo al momento de hacer pruebas para ver que todo el funcionamiento de la base de datos es correcto y coherente, especialmente cuando se trata de graficar rutas.

Desarrollar solamente el Frontend era una opción, pero la intención de esta tesis, además de conseguir el grado de Ingeniero de Sistemas, es la de dar desarrollar una herramienta que potencialmente pueda ser puesta al servicio de la sociedad para ayudarla a movilizarse dentro de la ciudad del Cusco, por lo tanto, el backend es necesario, y en consecuencia, es parte de la presente tesis.

4.1 Metodología de Desarrollo

Como se indicó al inicio, (véase 1.1.1) se utiliza el PUDS como metodología de desarrollo acompañado del marco de trabajo Scrum, ambos convergen en muchos aspectos, entre ellos el hecho que el desarrollo de software es iterativo e incremental. Sin embargo también tienen algunos puntos en los cuales opinan un poco distinto, como el caso de una de las frases del manifesto ágil, “Software por encima de documentación exhaustiva”. (véase 2.1.11)

El PUDS propone un desarrollo basado en 4 fases y 5 flujos de trabajo fundamentales (figura 5 en el punto 2.1.10), sin embargo también propone realizar diferentes tipos de diagramas y documentación, que en el caso de Scrum, no es obligatorio ni necesario. Por otro lado, Scrum propone un trabajo basado en historias de usuario y dividiendo los requerimientos en Sprints, para así lograr el desarrollo total del producto.

Al utilizar PUDS junto a Scrum; se busca seguir el proceso de desarrollo basado en las 4 fases de desarrollo de PUDS (Inicio, Elaboración, Construcción y Transición), pero acompañado de técnicas y artefactos de Scrum. Para tener todo esto más claro, se elaboró el siguiente listado donde se determina cómo se realizó el proceso de desarrollo utilizando PUDS y Scrum.

a) Se utiliza como base las 4 fases de desarrollo del PUDS, sin embargo estas se modifican de acuerdo a lineamientos de Scrum.
b) Se utiliza Scrum especialmente durante la fase de construcción.
c) No se utilizaron los casos de uso como herramienta de captura de requerimientos, sino historias de usuario.
d) El listado detallado de requerimientos se realizó utilizando la Lista de Producto en lugar de muchos diagramas de Casos de Uso.
e) No se realizó el modelado de negocios mediante herramientas propuestas por PUDS; en su lugar se utilizó el “Bussiness Model Canvas”.
f) Los 5 flujos de trabajo fundamentales (Requerimientos, Análisis, Diseño, Implementación, Prueba) son inherentes a cualquier proceso de desarrollo de software, por lo tanto son utilizados a lo largo del proyecto; mas no son documentados exhaustivamente.
g) El equipo de desarrollo no se conforma de la manera en que PUDS propone, sino como lo hace Scrum con los Equipos Scrum, para de esa manera integrar mejor a los clientes.

4.2 Fase de Inicio

Se debe tener en cuenta que la mayor parte de la fase de inicio se elaboró en el esquema de tesis, sin embargo, para evitar repetir información innecesariamente, el Modelo de Negocios se muestra en la fase de Elaboración, lo mismo la propuesta económica, y la arquitectura. Por otro lado, se explica a continuación elementos importantes propias de la fase de Inicio de este proyecto.

4.2.1 Delimitación del software

Se desarrolló un aplicativo para Android en el cual el usuario pueda ver las empresas registradas, rutas concesionadas (incluyendo gráficos de ida y vuelta), y que pueda guiar al usuario para que este pueda transportarse dentro de la ciudad del Cusco haciendo uso del servicio de transporte público de la ciudad, recibiendo las coordenadas de origen y destino del viaje; sugiriendo la(s) empresa(s) a usar para llegar a ese destino y los paraderos dónde subir y bajar, en caso no encuentre ninguna ruta; debe sugerir llegar a pie al destino. También debe graficar el camino desde el punto de origen al paradero de subida, la ruta que el bus va a seguir y el camino desde el paradero de bajada hasta el punto de destino.

Por otro lado, también se consideró necesario el desarrollo de un Sistema Web para mantener la base de datos, para que de esa manera se realicen todos los mantenimientos necesarios a partir de interfaces amigables. Finalmente, la parte de publicidad en la propuesta económica queda como plan para recuperar la inversión, ya que en la presente versión del producto todavía no está implementado.

4.2.2 Determinación de roles en el proyecto (Equipo Scrum)

Los roles dentro del proyecto han sido extraídos a partir de la conformación del equipo Scrum, (véase 2.1.12) por lo tanto mi persona es el equipo Scrum (dueño del producto, equipo de desarrollo y Scrum master), sin embargo; vale la pena dejar en claro la importancia de mi asesora como guía y puente entre el dueño del producto y el cliente final para el desarrollo del proyecto.

Al ser el dueño del producto, me hice cargo de generar las historias de usuario a partir de las necesidades propias del proyecto y las extraídas mediante encuestas. A partir de ese punto, como dueño del producto, se redactó la Lista de Producto.

Al ser el equipo de desarrollo, todas las tareas de desarrollo y planificación fueron hechos de manera personal, incluyendo claro está, la selección de requerimientos que se iba a realizar en cada Sprint. Al ser el Scrum master, todas las tareas propias del mismo fueron asumidas personalmente.

4.2.3 Definición de producto terminado

Una clave muy importante tanto para el Proceso Unificado de Desarrollo de Software como para Scrum, es tener definido el término de "Producto Terminado". En el presente proyecto el término se define como:

"Un producto será terminado cuando cumpla completamente con los objetivos que se presentaron al inicio del Sprint, sin embargo, dado que los requisitos pueden cambiar a medida que se avanza con el desarrollo, los productos terminados pueden ser desechados o modificados, dependiendo de la necesidad que se tenga"

4.2.4 Recopilación de historias de Usuario - Identificación de requerimientos

Las metodologías ágiles utilizan un artefacto en particular para la recolección de necesidades del software, este artefacto se llama "Historia de Usuario", y trata de lo siguiente; es un pedazo de papel donde un usuario del software escribe las distintas necesidades que tiene siguiendo una plantilla muy simple, la cual es: (Stellman A., Greene J.. 2015. pág.143)

<Nro de historia>
Como <aqui se ingresa el tipo de usuario> deseo <aquí se ingresa la acción específica que se desea que haga el software> para que <lo que deseo que suceda como resultado>.
<Agregar el valor de importancia del requerimiento>
(El valor se evalúa del 1 al 5, donde 1 es poco importante y 5 muy importante)

Como esta tesis tiene dos conjuntos de requerimientos, los del backend y los del frontend, las historias de usuario son distintas. A continuación se muestran las historias de usuario de ambos.

4.2.4.1 Historias de Usuario – Frontend:

Las Historias de Usuario en cuanto al frontend del proyecto (aplicación móvil), fueron hechas mediante dos fuentes. Como desarrollador del proyecto; las necesidades surgieron antes de inscribir la tesis, por lo tanto la idea ya era bastante clara desde ese entonces, sin embargo, estas se alimentaron a partir de las entrevistas realizadas de manera casual e informal en distintas partes de la ciudad, como conversando del tema con algún amigo o familiar, con la persona sentada al lado mientras se viajaba en un bus, o la persona que también esperaba en algún paradero. Las historias de usuarios seleccionadas se muestran a continuación en la tabla 4.

Tabla 4

Historias de Usuario del Frontend

Historia Nro. 1 – Frontend Como usuario del producto, deseo que la aplicación me permita ubicar el punto de origen y destino de mi ruta, para de esa manera indicar claramente desde dónde voy y hacia dónde voy. Importancia: 5
Historia Nro. 2 – Frontend Como dueño del producto, deseo que la aplicación localice a los usuarios automáticamente para ayudarlos a ubicar el punto donde se encuentra dentro de la provincia del Cusco, para que de esa manera seleccionar el origen sea más sencillo. Importancia: 5
Historia Nro. 3 – Frontend Como usuario del producto deseo que la aplicación me permita elegir la empresa que la que viajaré, para de esa manera tener poder de decisión. Importancia: 5
Historia Nro. 4 – Frontend Como usuario del producto, deseo poder ver la ruta que voy a seguir en la empresa que me embarqué, para conocer por dónde estoy viajando. Importancia: 4
Historia Nro. 5 – Frontend Como usuario del producto, deseo que la aplicación me indique cómo llegar al paradero que debo usar, y cómo llegar a mi destino desde el paradero que bajaré, para así saber qué paradero debo usar al subir y bajar. Importancia: 4
Historia Nro. 6 – Frontend Como dueño del producto, deseo que la aplicación indique al usuario su ubicación en tiempo real mientras este viaja en el bus elegido, para que en todo momento sepa dónde se encuentra y sepa cuánto le falta para bajar. Importancia: 5

Historia Nro. 7- Frontend Como dueño del producto, deseo que la aplicación permita a los usuarios visualizar las empresas existentes en el sistema, y las rutas que estas siguen tanto de ida como de vuelta, para que en cualquier momento pueda consultar las rutas, sus trazados y las empresas registradas. Importancia: 5

Fuente propia.

4.2.4.2 Historias de Usuario – Backend:

El backend tiene la función de agilizar todo el proceso de mantenimiento de la base de datos, los requisitos del mismo fueron extraídos tanto de la idea incial, como de las Historias de Usuarios generadas y los requerimientos que iban apareciendo a medida que se iba desarrollando. Fueron escritos y desarrollados personalmente, y se muestran en la tabla 5.

Tabla 5

Historias de Usuarios del Backend

Historia Nro. 1 – Backend Como equipo de desarrollo, deseo dar mantenimiento a las tablas de empresas de transporte mediante una interfaz amigable para evitar tener que colocar sentencias Sql cada vez que sea necesario. Importancia: 5
Historia Nro. 2 – Backend Como equipo de desarrollo, deseo dar mantenimiento a los paraderos registrados mediante una interfaz amigable, para evitar tener que hacerlo mediante sentencias Sql. Importancia: 5
Historia Nro. 3 – Backend Como equipo de desarrollo, deseo dar mantenimiento a las rutas y a sus coordenadas mediante una interfaz amigable, para evitar tener que hacerlo mediante sentencias Sql. Importancia: 5
Historia Nro. 4 – Backend Como equipo de desarrollo, deseo poder visualizar gráficamente el trazado de las rutas a medida que se van ingresando coordenadas, para darme cuenta rápidamente si el trazado que se va realizando es el correcto. Importancia: 5
Historia Nro. 5 – Backend Como equipo de desarrollo, deseo mantener una tabla donde se ingrese ordenadamente el orden de los paraderos a seguir de cada ruta ingresada (de ida y de vuelta), todo ello de manera gráfica, para poder indicar por cuáles paraderos pasa la ruta de ida y vuelta, y además para que en el frontend pueda realizar la ubicación de los paraderos cercanos a los puntos de origen y destino, y otras funciones propias del frontend. Importancia: 5

4.3 Fase de Elaboración

La fase de elaboración es la segunda fase del PUDS; durante la misma se realizo lo siguiente:

4.3.1 Arquitectura de Software

La Arquitectura de Software propuesta en la presente tesis es de “Cliente – Servidor”, se utiliza un servidor conectado a internet que se ubica en el servicio de hosting, dentro del cual se programa todo el backend; incluyendo los servicios web. Este servidor a su vez utiliza los mapas y servicios web de Google, y de esa manera es que se programa el backend.

En el caso del frontend, en la presente versión del producto, los dispositivos móviles con Android se conectan al servidor ubicado en el hosting a través de internet y hacen uso de los servicios web programados para hacer una copia a las tablas necesarias de la base de datos; e ingresarlas en la base de datos propia del móvil, que en este caso es SQLite. En la presente versión del producto es todo lo que hacen con el hosting, ya que después al tener los datos almacenados, se conectan a Google para hacer uso de los mapas y servicios web. En la figura 11 se detalla lo explicado.

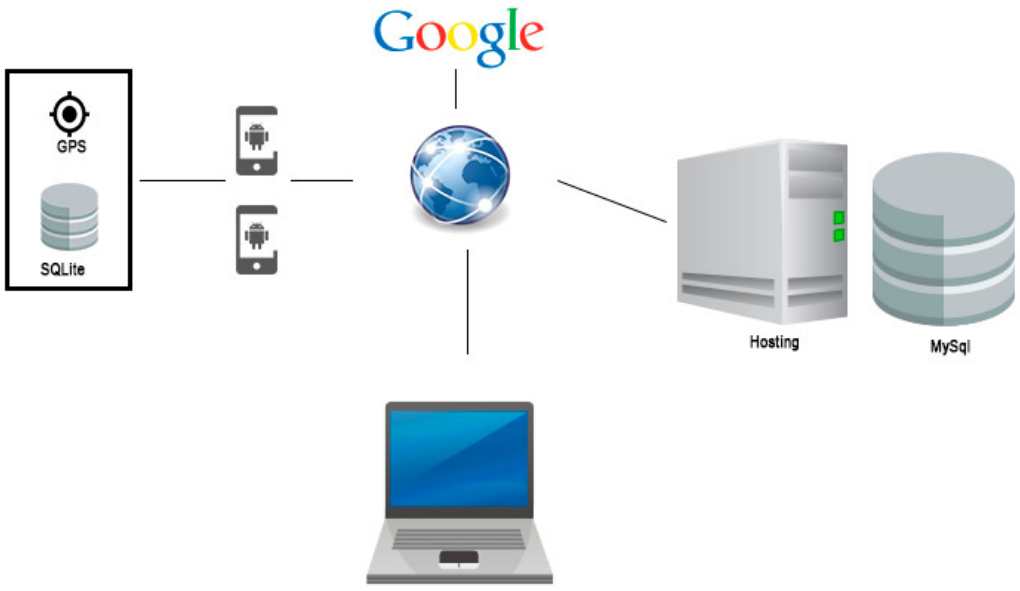

Figura 11. Arquitectura del Software. Fuente propia

4.3.2 Propuesta Económica

La propuesta económica consta de dos puntos importantes, coste de inversión en el software y la manera en que este generará ingresos.

La tabla 6 representa la propuesta económica. La moneda utilizada es la peruana (Sol), y los costos son estimados.

Tabla 6

Propuesta económica

Egresos	S/.	Ingresos	S/.
Recursos Humanos		Publicidad en aplicativo. S/.250.00 al mes (aprox 5 anuncios al mes)	1250
Salario de Desarrollador por 8 meses. S/.2500.00 al mes	20000.00		
		Pagos de empresa por derecho de listarse antes: S/.350.00/mes. (aprox. 4 empresas al mes)	1400
Equipos de cómputo			
- Laptop Lenovo Ideapad z500 Procesador core i5, 6gb ram, disco duro 1tb.	2000.00		
- Pantalla LG 19’	250.00	Donaciones Paypal	
- Mouse microsoft inalámbrico	50.00		
- Teléfono móvil Motorola G5	899.00		
Licencias de Software			
- Ninguna, todo es software libre. (Ubuntu, Libre Office, SQLite DB Browser, Netbeans, PHP, HTML, CSS, Javascript, XML, Android Studio, Java, Sqlite, MySql, Workbench, Gimp, Pencil, Star Uml, Firefox, Chrome, etc.)	0.00		
Otros			
- Servicios de Luz por 8 meses, S/.20.00 mensual	160.00		
- Servicio de Internet de 6gb por 8 meses. S/.89.9 mensual	718.20		
- 1 millar y medio de papel bond A4	120.00		
- Gastos de impresión	200.00		
- Transporte	400.00		
- Hosting y dominio por 1 año.	350.00		
Imprevistos (5% del total)	1257.36		
Total:	**26404.56**	**Total:**	**2650**

Fuente. Platzi y trabajadores UAC

En el caso de pagos de empresa por derecho de aparecer primero, se refiere a que las empresas que deseen aparecer primero en la aplicación al

momento que esta sugiere empresas, deberán pagar un monto mensual. La publicidad en el aplicativo tiene un costo mensual.

Se prevee recuperar la inversión y empezar a generar ingresos a partir del 11vo mes aproximadamente. (en caso se consiga los 5 anuncios y las 4 empresas).

4.3.3 Listado y mitigación de riesgos

En todo proyecto que se lleva a cabo existen riesgos, estos pueden ser financieros, de recursos humanos o incluso desastres naturales. Identificar riesgos y planear contramedidas es una práctica promovida por el PUDS, por lo tanto a continuación se muestran los riesgos potenciales encontrados para el presente proyecto y sus contramedidas correspondientes. El listado se puede apreciar en la tabla 7.

Tabla 7

Listado y plan de mitigación de riesgos

Riesgo1: Fallo del computador
Descripción: En caso la máquina donde se programa sufra un desperfecto, ya sea de software o hardware. Por ejm: fallos en disco duro, fallos de sistema operativo, fallos de IDEs, etc.
Contramedida: Copias diarias, de archivos creados o modificados, en una memoria USB y en el Google Drive. Los archivos copiados corresponden a: • Documentos generados propios de la tesis, como documentos en PDF y archivos de Libre Office. • Copia de seguridad del backend. • Copia de seguridad de la App. • Copias de seguridad de la base de datos backend (datos y estructura), y de la base de datos de la App (estructura).
Riesgo 2: Desastres naturales en la ciudad
Descripción: Desastres naturales que pueden suceder en la provincia del Cusco, particularmente; terremotos.
Contramedida: Las mismas que en el riesgo 1.

Riesgo 3: Retrasos en las entregas y avance del proyecto. **Descripción:** Retrasos al momento de programar debido a múltiples situaciones, como por ejemplo enfermedad o trabajos inesperados. **Contramedida:** Análisis y toma de decisiones en cuanto a reducción de objetivos a desarrollar para la App. Se evalúa los tiempos de entrega propuestos y dependiendo de si estos pueden extenderse o no, se procede a reducir algunos objetivos iniciales del proyecto, para así aligerar el proceso, teniendo en cuenta los menos importantes.
Riesgo 4: Aparición de otros programas similares en el tiempo de desarrollo **Descripción:** Mientras se va desarrollando, es posible que alguien lance al mercado un producto con características similares al que se va desarrollando. **Contramedida:** Visitar las empresas de transporte público para ir conociendo su interés en el producto que va en desarrollo, y a la vez saber si saben de alguien que esté haciendo lo mismo. Lanzamiento anticipado del producto. Dejar de lado cualquier oferta laboral que se presente para poder cumplir en el menor tiempo posible con la entrega.

Nota. Fuente Propia

4.3.4 Lista de Producto

La Lista de Producto es un compilado de todas las tareas que agregar valor al producto; y que faltan realizar. Este artefacto es usado generalmente para tener registro de todo lo que falta por hacer; y también como base para extraer las tareas a realizar en cada Sprint. La Lista de Producto es variable, y puede variar desde el inicio del proyecto hasta el final; dependiendo esto del Dueño del Producto.

En la tabla 8 se puede ver la Lista de Producto, la misma que contiene todos los requerimientos del backend y del frontend. La columna días indica el número de días que se demoró en desarrollar dicho requerimiento.

Tabla 8

Lista de Producto

Nro	Requerimiento	Días
	BACKEND	
	Sprint 1	
1	Investigación sobre uso de los mapas de Google.	2
2	Extraer coordenadas de puntos elegidos en el mapa.	1

3	Probar el gráfico de rutas.	3
4	Diseño del diagrama entidad relación	1
	Sprint 2	
5	Codificación básica de la interfaz y la hoja de estilos	4
6	Programación de las clases	2
7	Creación de la base de datos y sus usuarios	2
8	Creación de un loggeo para administrar el backend	2
	Sprint 3	
	Mantenimiento de paraderos	
	Agregar paraderos	
9	Diseño y programación de interfaz	1
10	Insertar ícono de paradero en mapa	1
11	Programar eventos a los íconos de paraderos	1
12	Agregar paradero en base de datos	1
	Modificar paraderos	
13	Diseño y programación de interfaz de elección y modificación.	1
14	Insertar íconos de paraderos en ambas interfaces	1
15	Programar eventos de íconos de paraderos	1
16	Modificar paradero en base de datos	1
	Eliminar paraderos	
17	Diseño y programación de interfaz	1
18	Programar eventos de los íconos de paraderos.	1
19	Programar mensaje de confirmación	1
20	Eliminar paradero de base de datos	1
21	Pruebas de mantenimiento de paraderos	7
	Mantenimiento de Rutas	10
	Agregar ruta	
22	Diseño y programación de interfaz	1
23	Ingreso de data, y puntos iniciales y finales en la ruta	1
24	Programar eventos en marcadores de punto inicial y final	1
	Agregar coordenadas de ida y vuelta	
25	Diseño y programación de interfaces	1
26	Programación ajax	1
27	Agregar coordenada y graficar ruta ingresada	1
	Modificar rutas	
28	Diseño y programación de interfaces	1
29	Selección de ruta y sentido con ajax	1
30	Insertar íconos en puntos de ruta	1
31	Programar eventos en marcadores de puntos de ruta	1

32	Modificar punto de ruta	1
	Mantener paraderos que pertenecen a rutas	
33	Diseño y programación de interfaces	1
34	Insertar paraderos e íconos que pertenecen a la ruta	1
35	Programar eventos en paraderos	1
36	Agregar o quitar paradero en ruta	1
	Eliminar rutas o coordenadas de rutas	
37	Diseño y programación de interfaces	1
38	Insertar íconos de ruta	1
39	Programar eventos en íconos	1
40	Eliminar ruta o coordenada de ruta	1
41	Pruebas de mantenimiento de rutas	9
	Mantenimiento de Empresas	
	Agregar Empresa	
42	Diseño y programación de interfaz	1
43	Ingreso de data de empresa	1
44	Asignación de ruta	1
	Modificar Empresa	
45	Diseño y programación de interfaz	1
46	Selección de empresa	1
47	Modificación de datos de empresa	1
48	Modificación de ruta concesionada de empresa	1
	Eliminar Empresa	
49	Diseño y programación de interfaz	1
50	Eliminación de empresa	1
51	Pruebas de mantenimiento de empresas	
	FRONTEND	
	Sprint 4	
52	Investigación sobre desarrollo en Android	5
53	Investigación sobre funcionamiento de XML	1
54	Investigación diseño y desarrollo de interfaces en Android	3
55	Investigación de mapas de Google para Android	2
56	Investigación de gráfica de rutas para mapas Google en Android	3
57	Investigación ingreso de íconos y marcadores en mapas Google para Android	1
58	Investigación de seguimiento mediante GPS en Android	3
59	Investigación de Servicios Web	3
60	Investigación de consumo de Servicios Web en Android	2
61	Investigación sobre SQLite	1