“DESARROLLO DE APLICACIONES MÓVILES EN EL SO ANDROID PARA LA DISMINUCIÓN DE LOS TIEMPOS DE RESPUESTA EN LAS SOLICITUDES DE SERVICIOS DE TAXI EN LA CIUDAD DE POPAYÁN, CAUCA, COLOMBIA”

MIGUEL EDUARDO ALEGRÍA LULIGO

YEYSON FERNANDO JIMÉNEZ URIBE

CORPORACIÓN UNIVERSITARIA AUTÓNOMA DEL CAUCA

FACULTAD DE INGENIERÍA

INGENIERÍA DE SISTEMAS INFORMÁTICOS

POPAYÁN

2016

“DESARROLLO DE APLICACIONES MÓVILES EN EL SO ANDROID PARA LA DISMINUCIÓN DE LOS TIEMPOS DE RESPUESTA EN LAS SOLICITUDES DE SERVICIOS DE TAXI EN LA CIUDAD DE POPAYÁN, CAUCA, COLOMBIA”

MIGUEL EDUARDO ALEGRÍA LULIGO

YEYSON FERNANDO JIMÉNEZ URIBE

DIRECTOR

JULIÁN DARÍO BERMÚDEZ TRUJILLO

PROYECTO PRESENTADO PARA OBTENCIÓN DEL TÍTULO EN:

INGENIERÍA DE SISTEMAS INFORMÁTICOS

CORPORACIÓN UNIVERSITARIA AUTÓNOMA DEL CAUCA

FACULTAD DE INGENIERÍA

INGENIERÍA DE SISTEMAS INFORMÁTICOS

POPAYÁN

2016

NOTA DE ACEPTACIÓN DEL DIRECTOR

El director y los jurados del trabajo de grado modalidad trabajo de investigación denominada: “DESARROLLO DE APLICACIONES MÓVILES EN EL SO ANDROID PARA LA DISMINUCIÓN DE LOS TIEMPOS DE RESPUESTA EN LAS SOLICITUDES DE SERVICIOS DE TAXI EN LA CIUDAD DE POPAYÁN, CAUCA, COLOMBIA”. Realizado por los estudiantes Yeyson Fernando Jiménez Uribe y Miguel Eduardo Alegría Lúligo, en modalidad trabajo de investigación, una vez revisado el informe final y aprobado la sustentación del mismo autorizan para que se realicen los trámites concernientes para optar por el título profesional de ingeniería de Sistemas

_____

Director de Proyecto Ingeniero Julián Bermúdez

DEDICATORIA

“pendiente”

AGRADECIMIENTOS

Damos gracias a todas las personas que nos han apoyado en esta etapa de la vida en la que nos formamos como profesionales y en especial a nuestras familias por brindarnos el apoyo necesario en los momentos más complicados de este largo y duro camino, por darnos la oportunidad de capacitarnos, por permitirnos terminar los estudios y apoyarnos en cada uno de nuestros proyectos emprendidos.

Iniciamos esta etapa como formación de ingenieros de sistemas con mucha incertidumbre de lo que nos pueda suceder en el camino pero con unas grandes ansias de formarnos como profesionales.

Agradecemos a todos los docentes que nos orientaron y apoyaron en este aprendizaje que se llevó con grande esfuerzo y sacrificio. Les damos nuestros más sinceros agradecimientos a los ingenieros Julián Bermúdez Trujillo y Jimmy Campo que gracias a su amplio conocimiento científico y su experiencia nos dieron como ejemplo de seguir adelante profesionalmente.

Nuestros padres les agradecemos por brindarnos la posibilidad de culminar nuestros estudios debido a que sin ellos no podríamos llegar a culminar estos, desde la primaria que nos educaron con valores como lo son humildad, responsabilidad, honradez, compromiso, dedicación, tolerancia, perseverancia y sobretodo con unas amplias ganas de salir adelante y ser profesionales.

Damos gracias a la Corporación Universitaria Autónoma Del Cauca por acogernos en su seno y darnos las herramientas necesarias y el apoyo durante todo el transcurso de la carrera profesional como Ingenieros de Sistemas Informáticos.

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

RESUMEN

Este proyecto fue desarrollado en conjunto con el proyecto titulado “Diseño e implementación del Backend para la gestión de aplicativos móviles del servicio de taxi en la ciudad de Popayán (Cauca, Colombia)”, que busca brindar una herramienta digital, para facilitar la toma del servicio de taxi en la ciudad. Tiene como fin la creación de las aplicaciones móviles para la plataforma Android, estas aplicaciones se enfocaron en conectar a los taxistas con la demanda de usuarios en la ciudad, de ahí que son dos aplicaciones, una para conductores y otra para usuarios con el fin de gestionar las peticiones del servicio.

Palabras clave: Movilidad, Android, Móvil, Transporte Terrestre individual terrestre, Servicios web, Internet, Scrum

ABSTRACT

This project is developed in partnership with the project entitled "Design and implementation of Backend for managing mobile applications of the taxi service in the city of Popayan (Cauca, Colombia)" Developed by Hector Satizabal, which seeks to provide a digital tool to facilitate decision taxi in the city. Aims to create mobile applications for the Android platform, these applications are focused on connecting drivers with user demand in the city, which is two applications, one for drivers and one for users to they can communicate with each other.

Key Words: Mobility, Android, Mobile, terrestrial Individual and Transport, web services, Internet, Scrum.

INTRODUCCIÓN

En la actualidad el transporte público individual de pasajeros es sumamente esencial, dado el estilo de vida que se lleva, pero ante la problemática de la alta demanda de transporte y poca oferta o problemas en el mismo, el transporte individual de pasajeros se ha deteriorado.

Por lo anteriormente expuesto este proyecto busca mejorar las necesidades del servicio de taxi para conductores y pasajeros, por esa razón es orientado a las plataformas móviles basadas en el sistema operativo Android ya que es el más usado en el mercado y permite llegar al mayor número de usuarios.

Este proyecto tiene como fin reducir los tiempos de respuesta en la confirmación del servicio de taxi, con el uso de la tecnología Sistema de posicionamiento global abreviado *GPS* e internet móvil, que son tecnologías de fácil acceso para la mayoría de las personas con el solo uso de teléfonos inteligentes.

El proceso utilizado para que el producto funcione será a través del consumo e integración de servicios web creados en el proyecto “Diseño e implementación del Backend para la gestión de aplicativos móviles del servicio de taxi en la ciudad de Popayán (Cauca, Colombia)” que permite la comunicación entre las aplicaciones del conductor y pasajero, el pasajero envía su petición de servicio con los datos necesarios (Ubicación, nombre teléfono, foto) y el conductor más cercano puede aceptar o declinar dentro de un límite de tiempo establecido, dependiendo de la disponibilidad de trabajo puede rechazar la petición, si es así pasara al conductor más cercano buscando la pronta respuesta del servicio.

Las herramientas móviles permitirán la petición del servicio de taxi, realizar un seguimiento en tiempo real al conductor y al finalizar la carrera calificar la experiencia del usuario con el servicio.

1. PLANTEAMIENTO DEL PROBLEMA

1.1 PLANTEAMIENTO DEL PROBLEMA

La ciudad de Popayán cuenta con 972 taxis[1] que están distribuidos en las siguientes empresas transportadoras: Servitaxi, Transtambo, Taxis Popayán y TaxBelalcazar, cada una cuenta con una cantidad de cupos asignados según su número de afiliados. Estas empresas prestadoras de servicio de transporte individual permiten a sus usuarios acceder al servicio por dos vías: la toma presencial del vehículo o mediante las líneas telefónicas de atención al cliente a través de su red privada de telefonía abreviado *PBX*. Se ha plasmado que los usuarios de este servicio no están satisfechos con los tiempos de respuesta[2], mediante la encuesta digital (Anexo A – Encuesta digital) realizada en la ciudad de Popayán por el equipo investigador se observa que el tiempo de llegada de un servicio de taxi en horas pico es superior a 15 min según el 60.8% de las personas encuestadas, por otro lado el 39.8% de las personas afirman que las zonas donde el servicio tarda en llegar es el norte.

El sistema actual es poco eficiente [2] al asignar las solicitudes del servicio, debido a que las líneas telefónicas utilizadas para atender las llamadas por *PBX* tienen un tiempo promedio de llegada de 10 a 15 minutos según la encuesta realizada[2]. Además en las horas pico se congestiona[2], en consecuencia al no poder solicitar el servicio y que las personas opten por “tomar el taxi en la calle”[3] acción potencialmente insegura y que supone un riesgo tanto para los pasajeros como para los conductores de transporte público individual (Taxi) quienes desconocen cualquier información de quien transportan, de ahí que el 86.9% de las personas que realizaron la encuesta estarían dispuestos en utilizar una aplicación móvil para la solitud del servicio.

Además, los usuarios no tienen la posibilidad de conocer los datos del conductor ni las placas del vehículo que presentará el servicio de transporte individual. También se presenta incertidumbre en la confirmación del número de orden del

servicio a prestar por diferentes inconvenientes como las grandes distancias entre el conductor y usuario, generando así un retraso en los tiempos de llegada y confirmación del estado de la solicitud del usuario, problema que se pretende resolver a través del desarrollo de este proyecto.

Otra dificultad que generan las líneas telefónicas al solicitar un servicio de taxi, es la incertidumbre sobre la aceptación de la solicitud, esto obliga a los clientes a realizar varias llamadas para conocer el tiempo de llegada de su taxi provocando una pérdida valiosa de tiempo o en el peor de los casos, se corre el riesgo que el servicio no llegue.

1.2 JUSTIFICACIÓN

1.2.1 Justificación técnica

En la ciudad de Popayán, tradicionalmente el servicio de taxi se ha realizado de la misma forma, por *PBX* o saliendo a la calle para conseguirlo, es por ello que los usuarios opinan que es un "pésimo servicio de taxis en Popayán" [3] debido a que sin beneficiarse de los recursos tecnológicos que en la actualidad existen no se lograría realizar una buena solución para este servicio. Herramientas como el internet, los dispositivos móviles, las bases de datos para registro y almacenamiento de la información, son algunas de ellas para suplir la necesidad de los usuarios y optimizar los tiempos de llegada.

Desde el punto de vista técnico se puede agregar que el problema en la aplicación de la empresa Servitaxi que tiene un comportamiento lento en los dispositivos, es debido a que esta fue desarrollada en un entorno de trabajo multiplataforma, en efecto no está optimizado para usar todas las características disponibles en los dispositivos móviles, así pues, es una oportunidad para crear las aplicaciones nativas para la plataforma Android, obteniendo como resultado los usuarios tengan una mejor experiencia. Para ello se usó Android Studio, Reactive Extencion Java

abreviado *RXJava*, la librería *Realm* como base de datos del dispositivo móvil y librería *retrofit* para la comunicación cliente servidor.

1.2.2 Justificación Funcional.

Gracias a las herramientas tecnológicas se logró reducir el tiempo de respuesta en la confirmación del servicio de taxi, ya que se implementó el sistema de respuesta de peticiones a los vehículos más cercanos al usuario en un rango de 5 kilómetros, obtenida mediante la ubicación GPS del dispositivo móvil del conductor, permitiendo asignar el servicio solo al primer conductor que acepte el servicio, así pues la distancia a recorrer será la más corta posible beneficiando al medio ambiente y al usuario en el tiempo de llegada.

Mediante los aplicativos móviles para la plataforma *Android*, se plantea reducir los tiempos de confirmación del servicio de taxi, permitiendo al usuario realizar la, visualizar los datos del conductor y el vehículo, seguimiento del vehículo por GPS y calificación del servicio.

1.2.3 Justificación social.

Debido a las necesidades de transporte y quejas por las condiciones que los usuarios describen, se hace evidente la necesidad de una solución que permita la pronta respuesta a la petición de un servicio de taxi desde una plataforma móvil.

Es por ello que se puede considerar una buena solución para mejorar el transporte público individual con la implementación de un sistema de petición y aceptación de taxis que garantice el buen uso y mejora del servicio. Al no depender de una empresa particular para la creación de este proyecto, las políticas internas de estas compañías no afectan en el desarrollo del mismo.

Se considera que los principales beneficiarios de la plataforma a desarrollar serán los ciudadanos, visitantes y turistas de la ciudad de Popayán al contar con una

aplicación móvil para la plataforma *Android* que les permitirá la solicitud del servicio de taxi desde el lugar donde se encuentren de una manera rápida y cómoda, las aplicaciones se ajustan acorde a la configuración de idioma inglés o español del dispositivo, el mayor beneficio para la comunidad será la pronta confirmación del servicio a diferencia de los medios tradicionales disponibles.

1.3 OBJETIVOS

1.3.1 Objetivo general.

Desarrollo de aplicaciones móviles que disminuya el tiempo de respuesta en las solicitudes de los servicios de taxi en la ciudad de Popayán Cauca, Colombia.

1.3.2 Objetivos específicos.

- Diseñar las aplicaciones con el uso de las recomendaciones y especificaciones *Material Design* de usabilidad.
- Realizar la implementación de las aplicaciones UniTaxi (Usuarios) y UniTaxi manager (Conductores) para la plataforma Android.
- Desarrollar un caso de estudio con un grupo de taxistas y usuarios del servicio de taxis en la ciudad de Popayán para evaluar las aplicaciones propuestas.

2. MARCO TEÓRICO

Teniendo en cuenta, el proceso que debe tener todo trabajo de investigación desde el momento en que se describe la necesidad, hasta obtener el producto exitosamente terminado. Es necesario hacer referencia a ciertos aspectos importantes que en el proceso han sido clave en el desarrollo del software, como lo son procedimientos, documentación y programas utilizados.

2.1 ANTECEDENTES

Los antecedentes tomados como referencia para abordar una solución software para la pronta respuesta a la petición del servicio de taxi fueron:

2.1.1 Easy Taxi

Cuenta con aplicaciones para Android, iOS y Windows Phone gratuitas[4], su última actualización fue el 20 octubre de 2016, y sus principales características son:

- Permite crear y modificar el perfil del usuario
- Integración con redes sociales
- Pagos en línea con tarjeta de crédito
- Visualización del vehículo en tiempo real
- Calificación del servicio

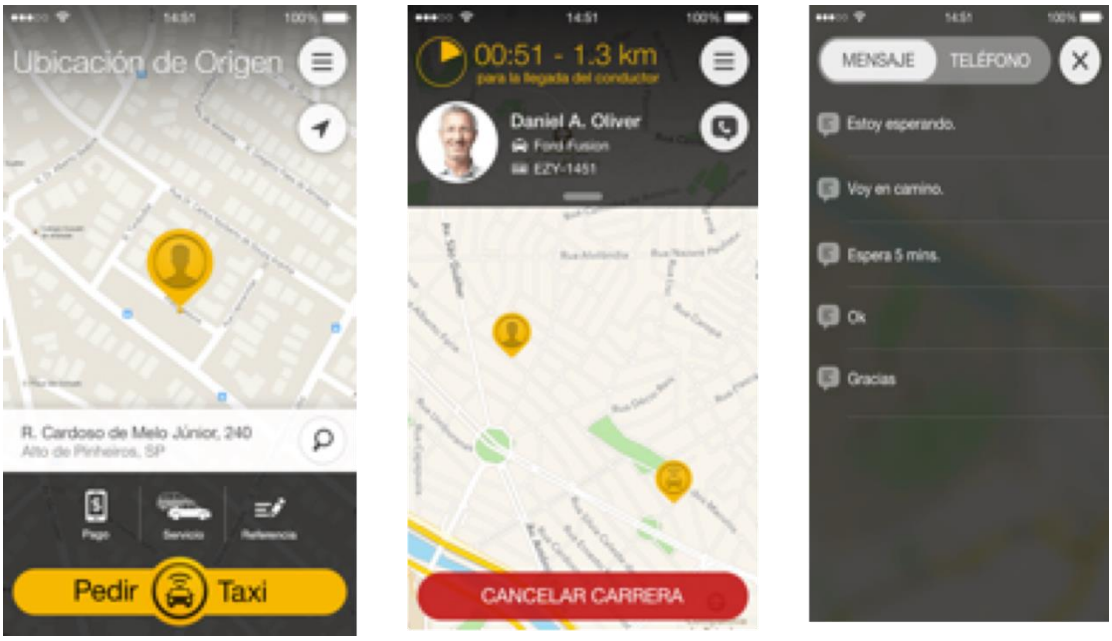


Figura 1. Interfaces de EASY TAXI

[4]

2.1.2 TAPPSI

Cuenta con aplicaciones para los Sistemas operativos Android, iOS, Windows Phone, FirefoxOS y BlackberryOS, además de una versión web[5] actualizado el 21 de octubre del 2016, sus principales características son:

- Gestión del perfil de usuario
- Seguimiento en tiempo real del vehículo
- Formas de pago en efectivo y electrónicos
- Servicio de encomiendas
- Calificación del servicio
- Chat con el conductor

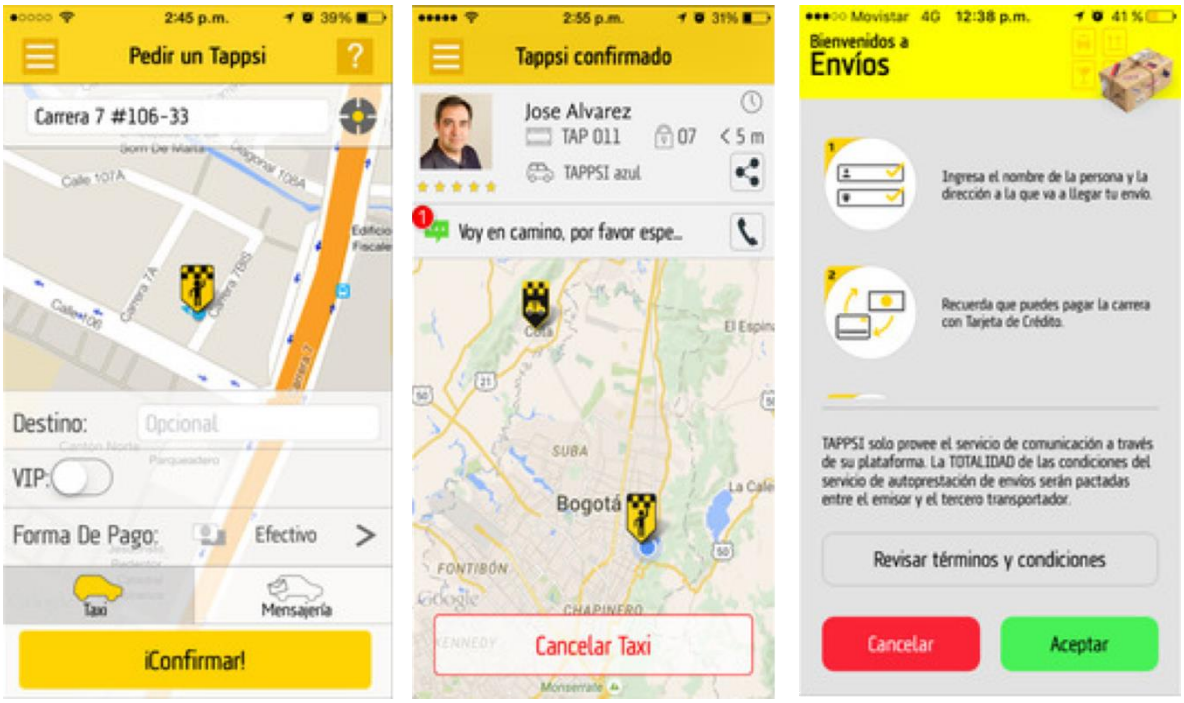


Figura 2. Interfaces de TAPPSI

[5]

2.1.3 SMARTTAXI

Cuenta con aplicaciones para los Sistemas operativos Android, iOS, además de una versión web[6], actualizado el 25 de octubre de 2016, sus principales características son:

- Creación y/o modificación perfil usuario
- Seguimiento en tiempo real del vehículo
- Chat con el conductor
- Calificación del servicio

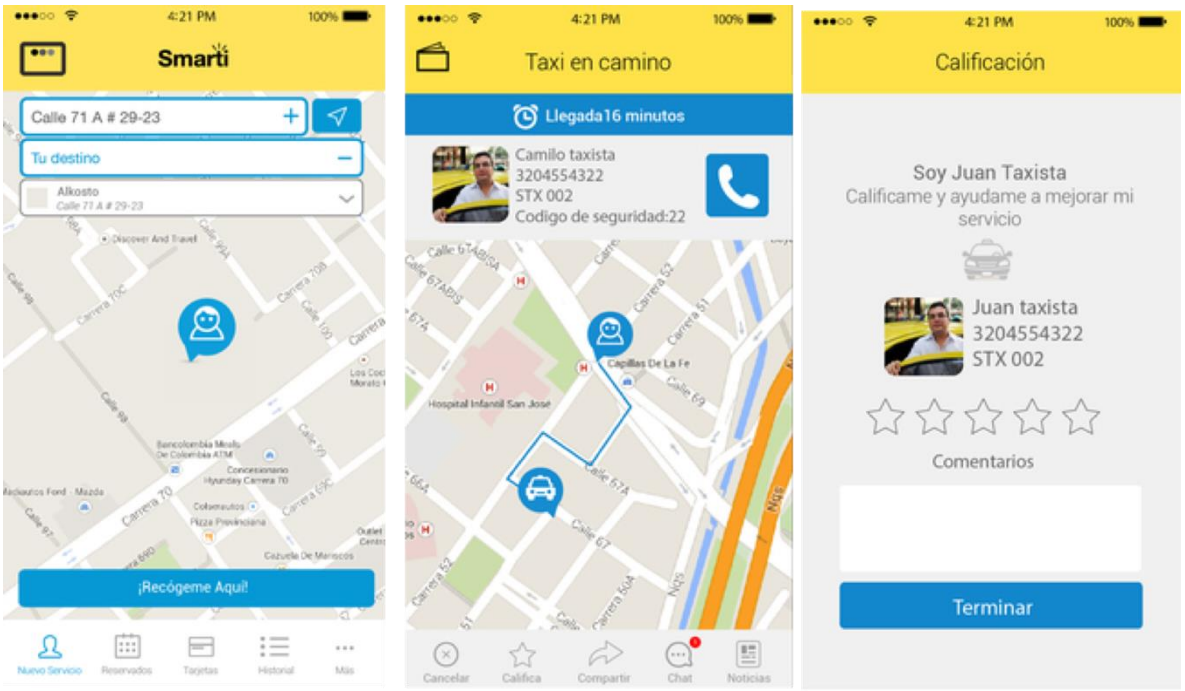


Figura 3. Interfaces de SMARTTAXI

[6]

2.1.4 CITYTAXI

Cuenta con aplicaciones para los Sistemas operativos Android, iOS[7], actualizado el 26 de octubre de 2016, sus principales características son:

- Creación y/o modificación perfil usuario
- Seguimiento en tiempo real del vehículo
- Historial de viajes
- Calificación del servicio

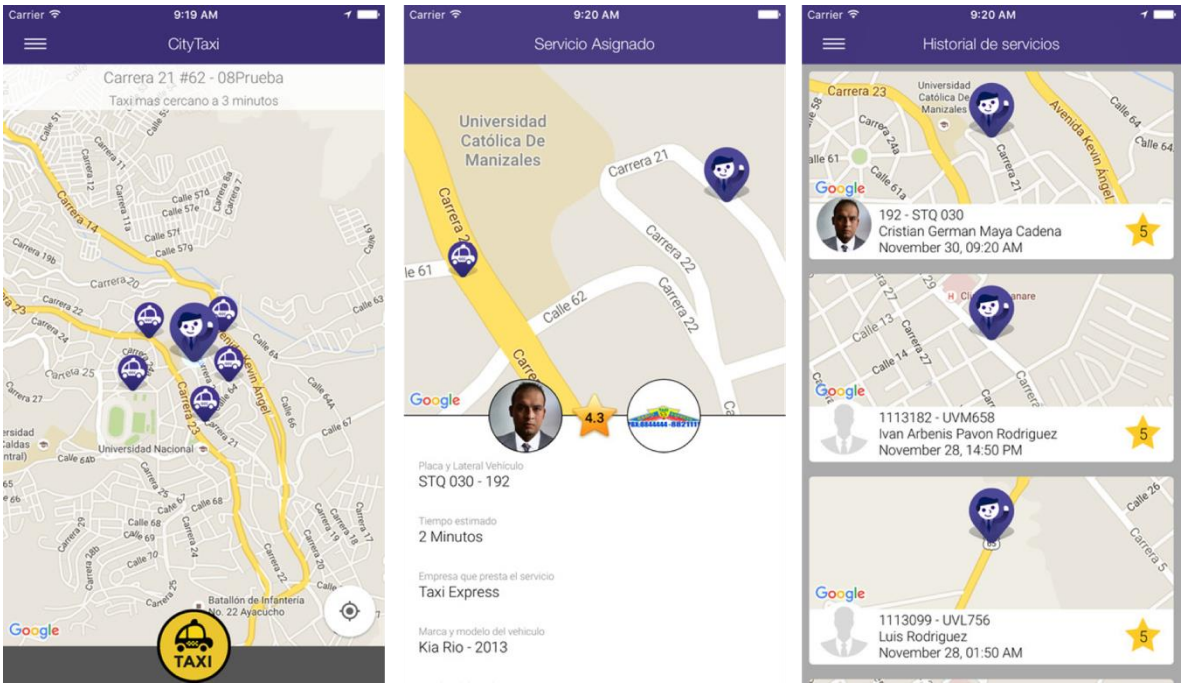


Figura 4. Interfaces de CITYTAXI

[7]

2.2 BASES TEÓRICAS

2.2.1 Plataforma Android

"Android es un sistema operativo basado en *Linux*, fue inicialmente desarrollado en el año 2005 por la empresa Android Inc., empresa que el gigante tecnológico Google adquirió ese mismo año"[8]. "Android fue presentado públicamente en 2007 en conjunto a la fundación del *Open Handset Alliance"*[9], un consorcio de empresas liderados por Google dedicada a la creación de estándares abiertos para dispositivos móviles, el primer resultado de esta alianza fue el dispositivo *HTC Dream* lanzado al mercado en octubre de 2008[10].

"El sistema operativo móvil Android se considera un grupo de tecnologías completo para dispositivos móviles, incluye el sistema operativo, aplicaciones

propias y de terceros”[11], esto ha causado que *Android* desde su salida oficial al público capte el interés de compañías fabricantes, desarrolladores y la audiencia en general, desde su inicio hasta la fecha *Android* no ha parado de crecer en audiencia, además de evolucionar en sus capacidades técnicas “soportando más *hardware*, como lo son las pantallas táctiles, sensores *GPS*, Comunicación de alcance cercano (abreviado *NFC)*, entre otros” [11]. Causando un gran impacto en los fabricantes que pueden adaptar sus diferentes dispositivos a este sistema operativo. Esto se debe en gran medida a que *Android* es de código abierto[11], esto significa que los fabricantes y desarrolladores lo pueden implementar sin ningún costo, respetando las licencias con las que fue creado.

“La arquitectura de *Android* consiste en un número de capas como, aplicaciones, entorno de trabajo de aplicaciones, librerías, *Android runtime* (tiempo de ejecución) y el núcleo de *Linux”*,[11]. La capa de aplicaciones permite a terceros el acceso a un set de características principales del dispositivo como el correo electrónico, envío de mensajes de texto, calendario, mapas, navegador de internet, entre otras, además de permitir el acceso a los sensores del dispositivo.

La capa de *Framework* de aplicaciones es un entorno de trabajo software por lo tanto permite que los desarrolladores implementen un estándar para la comunicación con el sistema operativo, con la ayuda de administradores, proveedores de contenido y otros servicios, permitiendo dar como resultado aplicaciones personalizadas para el sistema operativo, además en esta capa se presentan una serie de librerías escritas en leguaje C/C++, que pueden ser llamadas a través de una interfaz de java (*Java interface)*, permitiendo la creación de gráficos 2D, 3D, acceso a *Media Codecs* para el procesamiento de audio o video y el motor web *WebKit* para procesar elementos *HTML* Y *JavaScript*[11].

“Android runtime es un conjunto de librerías principales que proveen la mayor parte de las funcionalidades disponibles en la *Java interface.* Cada aplicación *Android* se ejecuta en su propio proceso ya que es una instancia de la máquina

virtual de java llamada *Dalvik virtual machine,* que transforma el código a lenguaje de máquina para que sea ejecutado" [11].

La arquitectura de Android (ver figura 11), *"*se base en *Linux* en su versión 2.6 para los servicios principales del sistema como la seguridad, manejo de la memoria, gestión del procesamiento, control de la red y los controladores. El núcleo de *Linux* es una capa de abstracción entre el hardware y el software"[11].

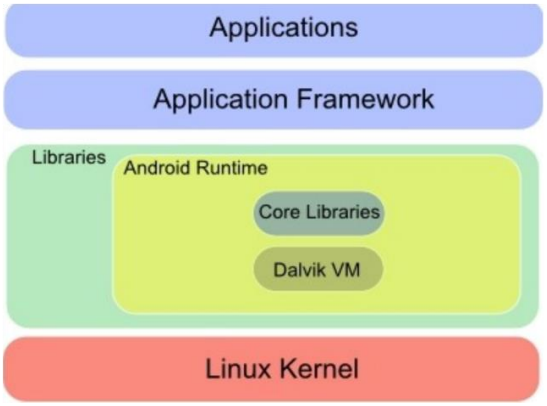


Figura 5. Arquitectura Android

[11]

2.2.2 RESTful Web Services

"La historia de *REST* (Transferencia de estado representacional) fue inicialmente descrito por Roy Fielding en su artículo "*Architectural Styles and the Design of Network-based Software Architectures"*"[12] en el año 2000, como un concepto *web* para la computación distribuida. Hoy en día se ha convertido en el estándar más popular usado por las compañías líderes como *Google* y *Facebook*, además ha sustituido gradualmente al estándar *Simple Object Acces Protocol* abreviado SOAP y los *Web Services Description Lenguage* abreviado *WSDL*, debido a que es una arquitectura más simple y ligera. El estándar *REST* y programas de *interface* de aplicación (APIs) son constantemente desarrolladas y adaptadas a los diferentes lenguajes de programación, por ejemplo para *Java* y Android existe la

librería *Retrofit* creada por la empresa *Square*[13], “siendo un cliente para Hypertext Transfer Protocol abreviado HTTP. Éste se considera un concepto de alto nivel que puede ser usado independiente de la plataforma o lenguaje, dada la naturaleza abstracta del concepto REST, no existe una especificación exacta acerca de sus métodos. Por ende el correcto estándar para la implementación de REST es HTTP y utiliza verbos HTTP (GET, POST, PUT, DELETE) para crear un estándar uniforme en su correcta implementación. La combinación de estos da como resultado la arquitectura RESTful Web Service basado en el uso de los cuatro métodos HTTP, Statelessness es decir el estado de la aplicación nunca se guardara en el servidor, las URLS se representan en un formato similar a los directorios en una computadora personal, los datos son transferidos en *eXtensible Markup Language* abreviado XML, *JavaScript Object Notation* abreviado JSON o en ambos formatos”[14].

2.2.3 Arquitectura RESTful

“Una arquitectura RESTful web service usa métodos http de acuerdo a la especificación RFC 7231”[15]. La característica clave de RESTful está en el uso explícito de los métodos HTTP, “por ejemplo http GET, es definido como un método de producción de datos que está destinado a ser usado por una aplicación cliente como un navegador de internet o una aplicación móvil, se usa para recuperar un recurso realizando una petición a un servidor web que procesará y entregará el conjunto de datos solicitados por el cliente.

REST le permite a los desarrolladores usar los métodos HTTP de una manera consciente en la definición del protocolo, el principio básico de REST permite realizar operaciones CRUD (crear, leer, actualizar y eliminar) definiendo los siguientes métodos”:[14]

- Para crear un recurso, POST
- Para recuperar un recurso, GET

- Para cambiar el estado de un recurso, PUT
- Para eliminar un recurso, DELETE

2.2.4 Métodos HTTP

2.2.4.1 HTTP GET

Este método aplica si la petición es ejecutada de manera satisfactoria. Cuando esto ocurre, GET retornara una representación en XML o JSON y la respuesta tendrá un código 200 que representa un OK[14], en el caso de fallar la petición responderá un 404, el famoso error NOT FOUND.

2.2.4.2 HTTP POST

Este método es usado principalmente para la creación de nuevos recursos, cuando una nueva entidad es creada, el servicio generalmente retorna en la respuesta un id que representa la forma de consultarlo en una petición GET, generalmente en forma de URL[14].

2.2.4.3 HTTP PUT

HTTP PUT es usado generalmente para actualizar algún recurso en concreto, deben encontrarse disponibles atreves de una URL, el cuerpo de la petición debe incluir los campos actualizados que ya se encuentran almacenados en el servidor, si la petición se ejecuta satisfactoriamente debe retornar un código de estado 200 y la respuesta no debe contener ningún elemento en su cuerpo con la intención de no sobrecargar la memoria[14].

2.2.4.4 HTTP DELETE

Esta operación se considera bastante sencilla, permite eliminar un elemento del servidor usando su identificador único, si el elemento fue removido exitosamente

HTTP DELETE retornara un 200 en su código de estado, este puede incluir en el cuerpo de la respuesta información detallada sobre el elemento eliminado[14].

2.2.5 Evolución de los Smartphone

Los sistemas operativos móviles [16] desde su aparición han ido avanzando muy rápidamente ya que van muy ligados a el avance de los dispositivos móviles así pues estos son los encargados de administrar los diferentes dispositivos o accesorios hardware que se les van implementando o agregando a dichos dispositivos como lo son Sistema Global Para comunicaciones Móviles (GSM), Sistema de Posicionamiento Global (GPS), Wireless Fidelity (WIFI), Conexión inalámbrica de corto alcance (Bluetooth), Comunicación de alcance cercano (NFC), acelerómetro, barómetro, sensor de luz, giroscopio entre otros. Todos estos dispositivos los debe administrar los sistemas operativos cada vez mejorando su desempeño y velocidad es por eso que han surgido diferentes versiones a lo largo del tiempo con sus ventajas y desventajas buscando así suplir las necesidades de los usuarios[17].

De acuerdo al estudio realizado por IDC [17] se ha decidido enfocar el desarrollo del proyecto en la plataforma *Android* ya que es la de mayor uso en el mercado según este estudio, permitiendo llegar en un futuro a la mayor cantidad de usuarios.

2.2.6 Normativa de diseño Material

Esta es una normativa de diseño presentada por Google, inc. en la conferencia Google I/O del 2014, inicialmente lanzado para las aplicaciones de google en el sistema operativo *Android*, esta normativa hoy en día se puede utilizar tanto en la web como en cualquier plataforma moderna, móvil o de escritorio, se trata de conceptos enfocados en el diseño utilizado por *Android* y se basa en un diseño limpio en el que sobresalen las animaciones, transiciones, colores y las sombras

con una filosofía de ser adaptable a cualquier tipo de pantalla con la mayor sencillez posible[11].

El diseño material en dispositivos *Android* es posible sin el uso de librerías a partir de *Android* 5.0, pero google ha puesto a disposición de los desarrolladores las librerías de compatibilidad para que se pueda implementar en versiones anteriores a esta, además de existir diversas librerías de terceros para la web y diversas plataformas.

Su principal diseñador es el chileno Matías Duarte[18] , la principal filosofía de material es que sea una abstracción al mundo real de los materiales, "Donde cada pieza tiene su lugar y espacio donde las animaciones sean lógicas, los objetos se superpongan pero no pueden atravesarse el uno al otro y demás[18].

2.2.7 Metodologías Agiles

Las metodologías agiles son todas aquellas que cumplen con una serie de características, principios y cumplen con los cuatro (4) valores.

2.2.7.1 Valores Agiles

- Los Individuos y su iteración más que los procesos y herramientas.
- El software que funciona más que la documentación extensiva.
- La colaboración con el cliente más que la negociación de un contratos
- La respuesta al cambio más que el seguimiento de un plan.

Con estos valores, las metodologías agiles pretende aumentar la eficiencia de todas las personas que estén involucradas en el proyecto generando así una disminución en costos y tiempo, a continuación se nombraran algunas metodologías agiles.

2.2.7.2 Principios Agiles

- La prioridad más alta es satisfacer al cliente con entregas tempranas y continuas de software valorable.
- Bienvenido requisitos cambiantes, incluso en desarrollo avanzado. Los procesos ágiles utilizan el cambio para dar ventaja competitiva al cliente.
- Entregar software funcional frecuentemente, desde un par de semanas a un par de meses, con preferencia a la escala de tiempo más corta.
- Empresarios y desarrolladores trabajan juntos diariamente durante el proyecto.
- Construir proyectos con personas motivadas. Entregarles el entorno y soporte que necesitan y confiar que harán el trabajo.
- El método más efectivo y eficiente para transmitir información desde y hacia un equipo de desarrollo es la conversación cara a cara.
- El software funcional es la principal medida de progreso.
- Los procesos ágiles promueven el desarrollo sostenible. Los patrocinadores, desarrolladores y usuarios deberían ser capaces de mantener un ritmo constante de manera indefinida.
- La atención continua a la excelencia técnica y buen diseño mejoran la agilidad.
- Simplicidad – el arte de maximizar la cantidad de trabajo no hecho – es esencial.
- Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.
- En intervalos regulares, el equipo reflexiona sobre cómo llegar a ser más eficaz, para luego ajustar su comportamiento de manera correspondiente [28].

2.3 METODOLOGÍAS DE DESARROLLO DE SOFTWARE

2.3.1 AUP

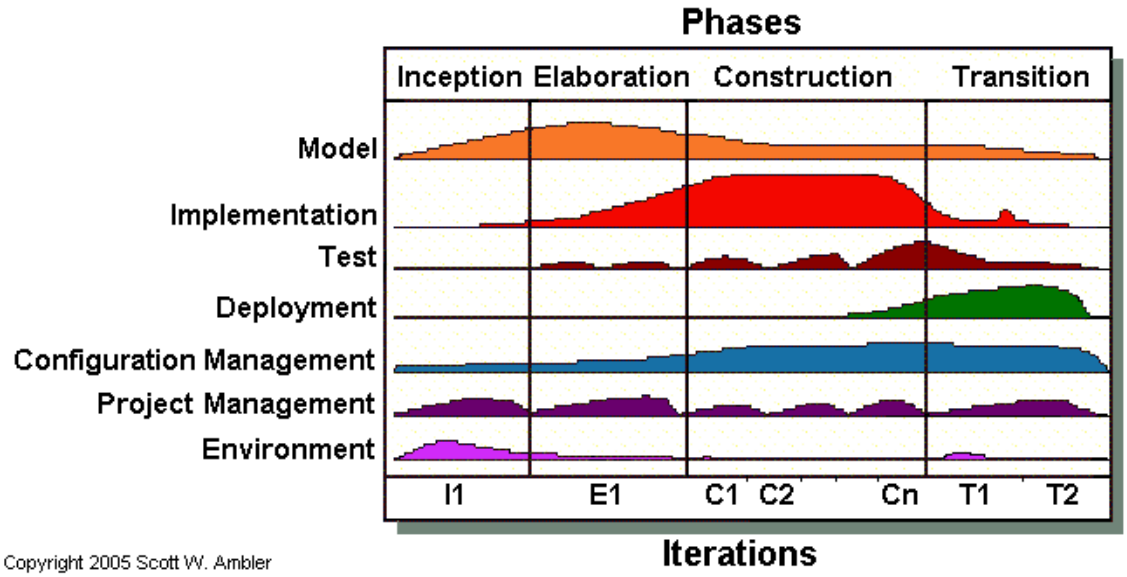


Figura 6. Ciclo de vida metodología AUP

[19]

Agile Unified Process abreviado AUP, es una versión simplificada de RUP (Proceso Unificado Racional) que busca enfocarse en técnicas agiles que se centran en las personas y en los resultados reduciendo el ciclo de vida de todo el proyecto esto lo realiza en cortos lapsos de tiempos llamados iteraciones estas están compuestas de una a cuatro semanas[20].

2.3.2 CRISTAL

Es una metodología de desarrollo de software ágil que consiste en calificar con un color cada tarea o método del sistema dependiendo de su complejidad y consecuencias graves que puedan llegar a ocurrir a lo largo del desarrollo[21].

2.3.3 LEAN Development

Es una metodología de desarrollo ágil adaptada del método de producción de la empresa Toyota se crea con un grupo pequeño de programadores altamente motivados con el fin de agilizar procesos y reducir costos y tiempo cada vez más debido a que se lleva en un constante aprendizaje del equipo[22].

2.3.4 Extreme Programming

Extreme Programming es una metodología ágil, que permite a los desarrolladores responder con seguridad a los requerimientos cambiantes del cliente, incluso cuando el proyecto ya está avanzado. Se basa principalmente en el trabajo en equipo, es decir los gerentes, clientes y desarrolladores son todos socios por igual en un equipo colaborativo, además el equipo debe ser auto organizado alrededor de problemas o las necesidades del proyecto, para resolverlo de la manera más eficiente posible[23].

Extreme Programming define cuatro variables para la estimación de un proyecto software **costo, tiempo, calidad y alcance**. Se establece que de estas cuatro variables, solo tres pueden ser establecidas a criterio por las personas externas al equipo de desarrollo y el valor de la variable restante se le asignara al equipo de desarrollo, por ejemplo si el cliente determina el alcance y la calidad, el gerente establece el precio, el equipo de desarrollo tendrá la libertad para determinar el tiempo de duración del proyecto. Este modelo fue propuesto por *Kent Beck* que demuestra las ventajas de esta metodología[24].

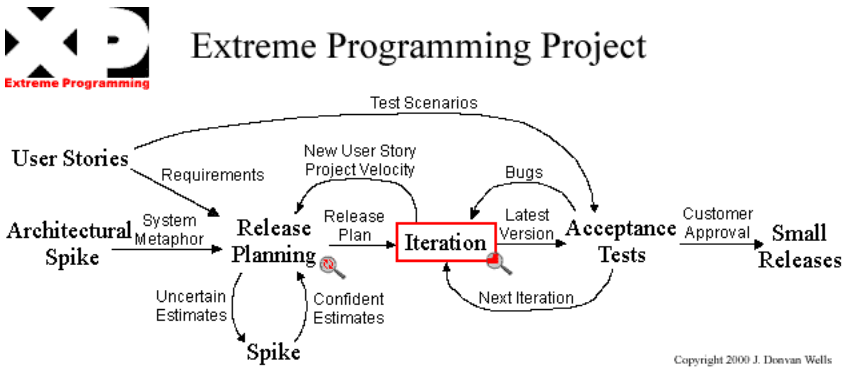


Figura 7. Extreme Programming Release Planning.

[24]

2.3.4.1 Fases extreme programming

Un proyecto gestionado con *Extreme Programming* es separado en una serie de fases:

- **Fase planificación**

En esta fase se requiere que se mantenga una comunicación constante de todo el equipo con el fin de entender y llegar a un acuerdo claro y conciso del alcance del proyecto, esto se lleva a cabo mediante herramientas que facilita el entendimiento de este, como lo son historias de usuario que son aquellas donde el cliente especifica los requisitos y funcionalidades del sistema lo más simple y posible[25].

- **Fase Diseño**

En este punto se elabora el sistema lo más simple y sencillo posible esto se puede llevar a cabo mediante prototipos o tarjetas de colaboración que especifican las responsabilidades y colaboradores de cada clase cumpliendo con todo lo requerido generando así la mejor solución para la satisfacción del cliente[25].

- **Fase Codificación**

Se desarrolla las funcionalidades del sistema que se especificaron en las historias de usuario esto se lleva a cabo en programación en parejas en un mismo computador esto se realiza con el fin de una continua integración y reducción de tiempos agilizando así la entrega del producto o funcionalidades[25].

- **Fase de pruebas**

Se realizan pruebas unitarias de todas las funcionalidades del sistema una por una todo esto se lleva a cabo de la mano del cliente y aceptación de este con el fin de llegar a una versión entregable[25].

2.3.4.2 Reglas y prácticas de Extreme Programming

Extreme Programming cuenta con conjunto de reglas y prácticas aplicadas a cada una de sus fases.

- **Historias de usuario**

“Estos documentos sustituyen a los documentos de especificación funcional y a los casos de uso de las metodologías clásicas, estos son escritos por el cliente, en su propio lenguaje describiendo lo que el sistema debe realizar.”[25] Estos documentos deben tener el mayor detalle para que el equipo de desarrollo pueda estimar sin riesgos el tiempo para esa iteración, si hay dudas los desarrolladores podrán dialogar directamente con el cliente para obtener los detalles necesarios.

- **Plan de entregas**

“El cronograma de entregas establece que historias de usuarios estarán agrupadas en un entregable y el orden de estos. Este cronograma será el resultado de una reunión entre todos los interesados del proyecto”[25] el cliente ordenara las historias de usuario según sus prioridades. Este se realizara de acuerdo a las estimaciones de tiempo diseñadas por el equipo de desarrollo.

Luego de algunas iteraciones es recomendable hacer nuevo el plan de entregas y ajustarlo de ser necesario

- **Plan de iteraciones**

“Las historias de usuarios seleccionadas para cada entrega son desarrolladas y probadas en un ciclo de iteración, de acuerdo a un orden preestablecido acorde a las prioridades dadas por el cliente. Al comienzo de cada ciclo, se realiza una reunión de planificación de la iteración. Cada historia de usuario se convierte en una serie de tareas específicas de programación. Asimismo para cada historia de usuarios se establecen los criterios de aceptación”[25], Estas pruebas se realizan al final del ciclo, además también deben realizarse al final de los ciclos siguientes para garantizar que no se ha afectado con los cambios realizados

- **Reuniones diarias de seguimientos**

“El objetivo de tener reuniones diarias es mantener la comunicación entre el equipo, compartir problemas y soluciones”[25], Es necesario que estas reuniones sean muy cortas y de pie para que sean lo más productivas posible, puesto que se obtienen las falencias, aciertos, el uso adecuado de herramientas y las posibles mejoras.

- **Diseño Simple**

“Extreme Programming propone implementar el diseño más simple que funcione”[25], los equipos mayormente fallan por implementar funcionalidades que no corresponden a la iteración o que no se han pedido.

- **Soluciones “Spike”**

“Cuando aparecen problemas técnicos, o cuando es difícil de estimar el tiempo para implementar una historia de usuario, pueden utilizarse pequeños programas de prueba (llamados “Spike”), para explorar diferentes soluciones”[25] Estos programas son únicamente para probar o evaluar una solución, y suelen ser desechados luego de su evaluación.

- **Recodificación**

“Esta práctica consiste en escribir una parte del código de un software, sin cambiar su funcionalidad, con el fin de hacerlo más simple, conciso y entendible. Esto se debe a que en muchas ocasiones cuando un desarrollador termina de escribir un código, este piensa que si lo comenzara de nuevo lo hubiera hecho de una forma diferente más claro y entendible, sin embargo como la funcionalidad ya está hecha esta práctica no se realiza. Extreme Programming sugiere que esta práctica de reescribir funcionalidades sea implementada cada vez que sea posible, esto se hace con el fin de mantener el código lo más simple posible”[25].

- **Disponibilidad del cliente**

“Uno de los requerimientos de Extreme Programming es tener al cliente o un representante del mismo, durante todo el proyecto, formando parte del proyecto, apoyando y aclarando dudas de los desarrolladores. El involucramiento del cliente es fundamental para la correcta implementación de Extreme Programming”[25]. Al estar el cliente en todo proceso del desarrollo del proyecto, puede prevenir situaciones no deseables o de funcionamientos alejados a la realidad que espera, dando como resultado un proyecto satisfactorio.

- **Programación dirigida a pruebas**

En las metodologías clásicas el módulo de pruebas es realizado generalmente al final del proceso de desarrollo otorgando un tiempo nulo o mínimo para la corrección de los errores encontrados en las pruebas, por ello “Extreme Programming promueve un modelo inverso que primero establece las pruebas que el sistema debe pasar. Las pruebas a las que se refiere esta práctica son los las pruebas unitarias realizadas por los desarrolladores”[25], con el fin de dirigir el desarrollo a la calidad estimada.

- **Programación en pares**

“Esto se compone de equipos de dos personas trabajando en un computador en la codificación del programa, mientras uno se ocupa de los detalles del código, su escritura, el otro mantiene la mirada en el contexto, revisa el código, su calidad y sugiere cambios.”[25] Esto crea un bucle de continua retroalimentación buscando que los errores sean mínimos. Aunque se cree que una programación en pares será más costosa, eso se compensa con menor inversión en horas y una mejora en la calidad del código.

- **Integración continua**

“Todo el equipo de desarrollo necesita trabajar siempre con la última versión del código,”[25] esto quiere decir que todas las mejoras y cambios realizados en las iteraciones anteriores deben incluirse en el desarrollo de la iteración actual. Extreme Programming promueve esta práctica con el fin de evitar futuros problemas de integración con versiones antiguas del código.

- **Pruebas unitarias**

“Las pruebas unitarias son una de las bases de Extreme Programming. Ya que todos los módulos realizados, deben superar las pruebas unitarias antes de poderse considerar finalizada la iteración, como se ha mencionado anteriormente las pruebas unitarias deben definirse antes de empezar con la codificación.”[25].

- **Detección y corrección de errores**

“Cuando se encuentra un error (“bug”), éste debe ser corregido inmediatamente, y se deben tener precauciones para que errores similares no vuelvan a ocurrir. Asimismo, se generan nuevas pruebas para verificar que el error haya sido resuelto”[25],

2.3.4.3 Valores de Extreme Programming

Extreme Programming se basa en cuatro valores estos deben estar presentes en los miembros del equipo, con el fin de que el proyecto se termine satisfactoriamente.

- **Comunicación**

"Los errores más comunes de en los proyectos software se deben a los problemas relacionados a la comunicación del equipo"[25], por ello Extreme Programming promueve una comunicación constante entre los miembros del equipo, al ser la documentación escasa una buena comunicación debe reemplazar la falta de documentación.

- **Simplicidad**

Extreme Programming apuesta por la sencillez en su máxima expresión, esta debe estar en la codificación, en los procesos a diseñar y realizar, "simplicidad es la máxima sofisticación"[26].

- **Retroalimentación**

La retroalimentación debe estar presente de forma permanente del lado de todos los interesados, cliente, gerentes y equipo de trabajo, de tal forma en la que todos aporten sus comentarios y se tomen decisiones para la siguiente iteración.

- **Coraje**

"Cuando se encuentran problemas serios en el diseño, o en cualquier otro aspecto, se debe tener el coraje suficiente como para encarar su solución, sin importar que tan difícil sea"[25],

2.4 MARCO DE TRABAJO DE GESTIÓN DEL PROYECTO

2.4.1 Scrum

Figura 8. Entorno de Trabajo SCRUM

[27]

Scrum es un entorno de trabajo para la gestión de cualquier tipo de proyecto, sean del tipo software o cualquier otro; pero dentro de este marco de trabajo se pueden emplear varias técnicas y procesos de acuerdo a la "*La guía de Scrum*" [25].

Scrum trabaja bajo 3 pilares fundamentales, que permiten el control de un proyecto, estos pilares son transparencia, inspección y adaptación.

Scrum es muy claro con sus actores y plantea solo 3, que hacen parte del Equipo Scrum (Scrum Team), dueño de producto (Product Owner), el equipo de desarrollo (Development Team) y el Scrum Master.

2.4.1.1 Pilares Scrum

Scrum se basa en los siguientes pilares:

- Transparencia: “Conceptualización de los aspectos importantes y que sean visibles para el equipo de desarrollo” [28]
- Inspección: “Inspeccionar con frecuentemente los artefactos de *Scrum* para detectar variaciones y realizar correcciones a tiempo” [28]
- Adaptación: **“**Realizar ajustes a los problemas que se obtienen desde la inspección” [28]

2.4.1.2 Equipo Scrum

Figura 9. Equipo Scrum (Scrum Team)

[29]

El equipo Scrum está constituido por el Dueño de Producto llamado “Product Owner”, el Equipo de Desarrollo denominado “Development Team” y el “Scrum Master” [30].

- **Dueño del Producto (Product Owner):** Es el encargado de maximizar el valor del producto entregado, su principal función es la de controlar el Product Backlog, es por esto que se le conoce como el representante del interés del cliente dentro del equipo.
- **Equipo de Desarrollo (Development Team):** Son los encargados de entregar los incrementos del producto, estos profesionales tienen las siguientes características:

- Son auto organizado.
- Los Equipos de Desarrollo son multifuncionales.
- Scrum no reconoce sub-equipos en los equipos de desarrollo
- El equipo es responsable como un todo, no hay errores individuales.

- **Scrum Master:** Es un líder que debe estar siempre al servicio del equipo Scrum, este debe permitir la interacción entre los miembros del equipo y personas externas, debe lograr que toda la teoría, reglas y prácticas sean entendidas por todos los miembros.

2.4.1.3 Artefactos

- **Pila de producto (Product backlog):** "Lista de ítems que representan trabajo pendiente, de ahí que el *Product Owner* y el equipo de desarrollado determinan los ítems que serán desarrollados en el siguiente *sprint*, por tanto esta lista debe estar lo más actualizada posible"[28].

- **Pila de Sprint (Sprint backlog):** Sprint backlog es la lista de pendientes ajustada a un orden de ejecución de las características del sistema del sprint que está en ejecución, por tanto esta lista es estimada por el equipo de desarrollo.

- **Incremento:** "Son todas las características del sistema que pertenecen a un sprint que son completados al final de este. Así pues al final este nuevo incremento debe estar con un estado de "Terminado" [28]

2.4.1.4 Eventos

- **Sprint:** “Es el corazón de Scrum con durabilidad de un mes o menos, en él se define que se construirá, el diseño, un planificación de cómo será construido, el trabajo a realizar y el producto que deberá resultar al finalizar el sprint, de ahí que el sprint tiene unas características como no realizar cambios que puedan afectar al objetivo del *sprint,* el alcance puede ser clarificado y renegociado entre el dueño de producto y el equipo de desarrollo, entre otros ” [28]

- **Incepción Ágil (Agile inception):** Aunque oficialmente no hace parte de los eventos de Scrum, cada vez más son los equipos de trabajo que lo implementan y consiste en que “Al iniciar un proyecto las personas que están directamente implicadas en esté, deben clarificar las inquietudes obteniendo así una visión compartida del producto que se desea obtener. Así pues, para preparar una inception se deben realizar las siguientes preguntas, cada una resuelta a través de dinámicas de grupo, sesiones de juegos, buscando que sean siempre en un ambiente tranquilo y que motive el trabajo creativo :
  - ¿Porque estamos aquí?: Porque realizar el producto
  - Elevator Pitch: Cómo defender el proyecto.
  - Diseña Tu caja: Permite explicar los beneficios que tendrían las personas con el producto.
  - Crea tu not list: describe cuales son las funcionalidades que tendrá el producto, las que no están contempladas y las que quizás se deben discutir para ser realizadas.
  - Que te quita el sueño: Determinar miedos y riesgos para minimizarlos y evitarlos.
  - Calcula tu tamaño: Tiempo que se usará para el cumplimiento del alcance.

- Cuáles son tus prioridades: Indicar el alcance, presupuesto y tiempo[30].

- **Planificación del Sprint (Sprint planning):** “Reunión de Planificación de Sprint permite al equipo de desarrollo definir junto al Product Owner las funcionalidades que se van a realizar en el desarrollo del sprint, así pues en esta reunión se debe tener el Product backlog, el último incremento con estado terminado del anterior sprint, y el rendimiento obtenido del equipo de desarrollo, de manera tal que el equipo de desarrollo se comprometa y sea capaz de realizar determinados ítem de lista priorizada”[28]

- **Reunión Diaria (Daily meeting):** “Es una reunión con un bloque de tiempo de 15 minutos para que el equipo de desarrollo sincronice sus actividades y cree un plan para las siguientes 24 horas” [28].

- **Sprint Review (Revisión de Sprint):** También se le conoce como Demo. Al finalizar el sprint se realiza una revisión para la aprobación de la entrega formal del producto, también tiene como objetivo hacer una retroalimentación de las actividades que se realizaron durante el *sprint* [28].

- **Retrospective de Sprint (Sprint Retrospective):** “El propósito de la retrospectiva es evaluar ante todo el proceso del Sprint y se tiene en cuenta situaciones como:

  - Inspeccionar cómo fue el último Sprint en cuanto a personas, relaciones, procesos y herramientas.
  - Identificar y ordenar los elementos más importantes que salieron bien, que salió mal y las posibles mejoras” [30].

2.5 MARCO CONTEXTUAL

En la ciudad de Popayán, tradicionalmente el servicio de taxi se ha realizado de la misma forma a través de llamadas a un PBX de alguna de las empresas prestadoras del servicio o saliendo a la calle para conseguirlo. Es por ello que no se tiene un control de información de las personas que con frecuencia utilizan el servicio, acerca de los taxistas, ni sobre la seguridad de los usuarios ofrecida por las empresas de taxi al acceder a un servicio.

Debido a las necesidades de transporte, se hace evidente la necesidad de una solución que permita la pronta respuesta a la petición de un servicio de taxi desde una plataforma móvil, de ahí se considera una buena solución que mejore el transporte público individual con la implementación de un sistema de petición y aceptación de taxis que garantice el buen uso y mejora de este servicio.

2.6 HERRAMIENTAS DE APOYO

2.6.1 Adobe Illustrator

Este es un editor de gráficos en vectoriales propiedad de *Adobe Systems*, permite crear ilustraciones, logotipos, iconos, tipografías, entre otros, se pueden exportar de forma optimizadas en formato PNG, SVG u otros, para su implementación en sitios web, aplicaciones móviles, impresiones y otros.

Actualmente forma parte de la familia *Adobe Creative Cloud* y tiene como única función la creación de material gráfico, la extensión de sus proyectos es .AI. Actualmente su costo es de $35.000 COP al mes [31].

2.6.2 Android Studio

Este un el IDE (Entorno de desarrollo integrado) oficial para el desarrollo de aplicaciones para la plataforma *Android*, creado y mantenido por Google, cuenta con un editor de código potente que es compatible con leguaje Java, Gradle, C++ y otras herramientas, cuenta con integración a diferentes sistemas de control de versiones como Git, Un emulador de Android permite al desarrollador probar sus aplicaciones sin necesidad de un dispositivo real[32].

Android Studio está basado en la plataforma *IntelliJ IDEA* en su versión de código libre[33] y hereda la mayoría de sus características. Cuando se crea un proyecto en esta herramienta esta puede contener generalmente los siguientes módulos:

- Modulo APP para Android
- Módulo de bibliotecas
- Módulo de Google App Engine (Lado del servidor)

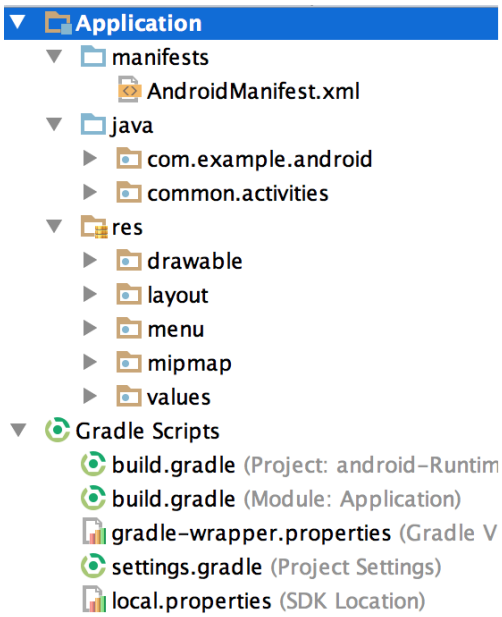


Figura 10. Estructura de carpetas proyecto Android Studio

[32]

2.6.3 Bitbucket

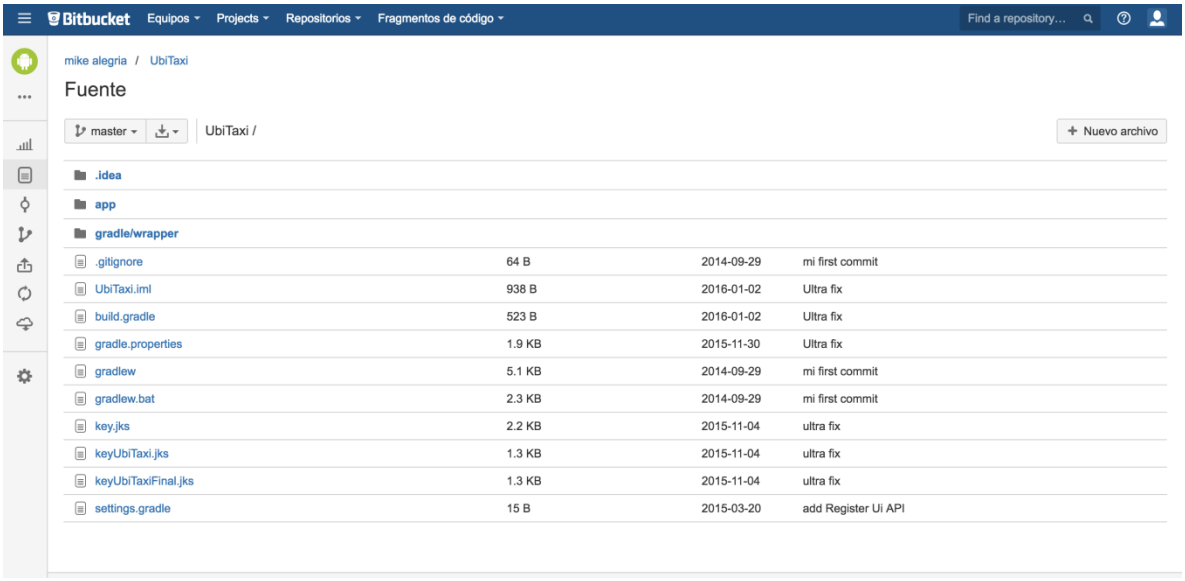


Figura 11. Bitbucket [Fuente Propia]

“Es una herramienta web basada en el control de versiones más popular llamado Git una de las grandes ventajas de dicha aplicación es su versión gratuita que permite manejar repositorios gratuitos y privados hasta un límite de 5 personas trabajando en el mismo proyecto”[34].

2.6.4 Firebase Cloud Messaging (FCM)

Figura 12. Estructura FCM

[35]

Firebase Cloud Messaging (FCM) es la evolución de *Google Cloud Messaging* bajo la marca *Firebase* propiedad de Google[35], esta es una solución multiplataforma que permite enviar mensajes y notificaciones, por ejemplo se le puede decir a una aplicación cliente hay nuevas actualizaciones de su correo electrónico u otros datos están disponibles para sincronizarse, con el único limitante de que los mensajes enviados no pueden superar los cuatro kilobytes (4KB)[36].

Estos mensajes pueden ser enviados desde el lado del servidor atreves de peticiones HTTP o XMPP hacia el servidor FCM que entregara los mensajes a los clientes indicados.

2.6.5 Git

Figura 13. Estructura de Git

[37]

Git es un software de control de versiones, diseñador originalmente por Linus Torvalds, basado en la experiencia propia de trabajar en el mantenimiento de una enorme cantidad de código gestionado por muchos programadores[38].

Git permite el trabajo paralelo de varios programadores en un mismo proyecto, integrando sus propias copias de trabajo en un repositorio centralizado, para su posterior integración con otras copias de trabajo o puestas en producción. Git se basa en una estructura de tres secciones, que son las siguientes:

- Directorio Git: Plataforma que guarda los objetos históricos del código.
- Directorio de trabajo: almacena los archivos actuales con los últimos cambios al proyecto.
- Índice: Este es un archivo que indica que información se va a enviar al directorio Git.

2.6.6 Planning Poker

“Es una técnica utilizada para dar estimación del peso de cada una de las tareas que se deben desarrollar a lo largo del proyecto esto se hace con una serie de cartas con números que comúnmente son la serie de Fibonacci” [47].

2.6.7 Trello

“Es una herramienta web que nos facilita un tablero colaborativo entre dos o más personas gestionando las listas de las tareas con el seguimiento más detallado de cada una de ellas con el fin de llegar a un estado para un mayor control del proceso del proyecto” [39].

Figura 14. Trello

[39]

2.7 GLOSARIO

- **Android:** Sistema operativo más utilizado para dispositivos móviles o Smartphone basado en Linux[11].

- **Api:** Interfaz o abstracción de funciones, procedimientos o métodos como una capa de abstracción para luego proceder a ser utilizada por otro software[40].
- **APP:** Sigla más utilizada para referirse a cualquier tipo de aplicación software en diferentes sistemas operativos[41].
- **Clientes:** Todo aquel dispositivo que se conecte o comunique con el servidor web de la plataforma.
- **Daily:** Cortas reuniones diarias al final de cada jornada de trabajo máximo de 15 minutos[28].
- **Desarrollo:** En esta fase los desarrolladores o ingenieros se dividen las funcionalidades del sistema para realizar la programación de las funcionalidades[28].

- **Development Team:** Todas las personas del equipo de desarrollo involucradas en el proyecto como lo son desarrolladores, diseñadores, Scrum Master y testers[28].
- **Diseño:** Flujo de trabajo del proceso que permite modelar la arquitectura y las funcionalidades del sistema[28].
- **Framework:** Entorno de trabajo con ciertas prácticas y funcionalidades previamente establecidas con el fin de facilitar y agilizar el proyecto para no empezar desde cero[42].
- **GPS:** Sistema de posicionamiento digital que permite ubicar un dispositivo en el mapa generando la latitud y longitud[43].
- **Iteración:** Es la repetición del ciclo de vida del Scrum esto se lleva a cabo al finalizar cada entregable o sprint[28].

- **JAVA:** Lenguaje de programación multiplataforma con el fin de desarrollarlos una vez para luego poder ser desplegados en cualquier plataforma[44].

- **JSON:** Un formato de texto ligero con una serie de parámetros con el fin del facilitar el intercambio y comunicación de datos entre distintas plataformas[45].
- **Linux:** Sistema operativo de software libre, es decir su código fuente puede ser modificado o vendido libremente[46].
- **Metodología:** Es una serie pasos o métodos previamente establecidos para cumplir un desarrollo o proyecto de la mejor manera posible.

- **Notificaciones PUSH:** Aquella comunicación de datos por medio notificaciones que proporciona google para el sistema operativo Android[47].
- **Peticiones HTTP:** Son un tipo de comunicación en la web para la transferencia de archivos o datos entre distintas aplicaciones[48].
- **Planificación:** Se plantea todos los aspectos de desarrollo método de trabajo tiempo y entregables que se va a desarrollar[49].
- **Product backlog:** Es la pila del producto una lista priorizada de las funcionalidades que debe tener el proyecto[28].
- **Product Owner:** Es el encargado de representar todos los intereses del cliente a lo largo del proyecto[28].
- **Scrum:** Es una guía o entorno de trabajo con una serie de pasos, reglas y roles para gestión cualquier tipo de proyectos agiles[28].
- **Scrum master:** Es aquella persona que sirve de facilitador y verifica que se cumplan el proceso de Scrum y debe velar por el equipo de trabajo tenga todas las condiciones necesarias para cumplir con el desarrollo del proyecto[28].
- **Sprint:** Son lapsos de tiempos no mayor a un mes por lo tanto se deben presentar entregas de avances funcionales desarrollados a lo largo del proyecto[28].

- **Stack:** Grupo de tecnologías para desarrollo de software que se comunican entre sí para dicho funcionamiento de un proyecto[50].
- **XML:** Lenguaje de marcas extensible es utilizado para guardar datos de forma legible con una estructura establecida con el fin de acceder fácilmente a ellos[51].
- **XP:** Metodología para desarrollo de aplicaciones o proyectos software agiles con una serie reglas y pasos a seguir[49].

3. METODOLOGÍA

3.1 IMPLEMENTACIÓN DE SCRUM

Figura 15. Entorno de Trabajo SCRUM

[27]

Soportados en el marco de trabajo ágil Scrum, se determinaron los roles, actividades y artefactos que se tendrán en cuenta en la implementación del proyecto.

3.1.1 Asignación de Roles

3.1.1.2 Scrum Team

Rol conformado por todos los integrantes del equipo es decir usuarios finales tanto taxistas como pasajeros también por el equipo de desarrollo y su respectivo líder cada uno con sus respectivas responsabilidades

3.1.1.3 Product Owner

Teniendo en cuenta las funciones del rol de Product Owner que Scrum propone, no se cuenta con una sola persona que cumpla con este rol es por ello que se

decidió que los integrantes del Scrum Team en conjunto asumirían estas funciones apoyados del director del proyecto de grado.

3.1.1.4 Scrum Master

Este rol fue asumido por uno de los miembros del equipo rotando la responsabilidad de este rol por cada sprint.

3.1.1.5 Development Team

Este rol fue conformado por el equipo de desarrollo, para el proyecto se asumieron papeles de desarrolladores, diseñador gráfico y tester.

3.1.2 Artefactos

3.1.2.1 Product Backlog

Se realizó la lista priorizada de las historias de usuario que deben ser implementadas al transcurrir el proyecto. La documentación y seguimiento se hizo utilizando la herramienta *Trello.*

3.1.2.2 Sprint Backlog

Esta actividad preparó las tareas por cada historia de usuario que sería implementada en cada *sprint,* esto se desarrolló de acuerdo a las prioridades previamente documentadas en *Trello y* estimadas con el apoyo de la técnica de *Planning póker*, que permitió tener un objetivo a entregar al finalizar cada Sprint.

3.1.2.3 Incremento

El desarrollo de las tareas e historias de usuario entregan al final de un Sprint, versiones funcionales que se convierte en una versión *reléase* que se compromete a entregar al finalizar el *sprint*.

3.1.3 Actividades

3.1.3.1 Planificación de Sprint (o Agile Inception)

- **Inception**

En esta parte se realizó la reunión previa con todos los interesados en el proyecto definiendo con exactitud la idea de lo que se va a desarrollar esto se llevó a cabo mediante una serie de pasos que fueron los siguientes.

- ¿Porque estamos aquí?
- Elevator pitch
- Diseña tu caja
- Que te quita el sueño
- Calcula un tamaño
- Cuáles son tus prioridades

3.1.3.2 Planificación Sprint 1

Para este sprint se hizo una estimación de 15 días que permitió ejecutar 4 historias de usuario incluido la aplicación móvil del lado del usuario y la aplicación del lado del conductor, de ahí que las historias de usuario estimadas son:

Aplicación Cliente:

- Registro de Usuario.
- Ubicación del taxi disponible.

Aplicación Taxis:

- Logueo de conductor.
- Estado del conductor.

3.1.3.3 Planificación Sprint 2

Para este sprint se hizo una estimación de 15 días que permitió ejecutar 4 historias de usuario incluido la aplicación móvil del lado del usuario y la aplicación del lado del conductor, de ahí que las historias de usuario estimadas son:

Aplicación Cliente:

- Solicitar servicio.
- Notificar ubicación del conductor

Aplicación Taxis:

- Aceptar servicio
- Notificar ubicación del usuario.

3.1.3.4 Planificación Sprint 3

Para este sprint se hizo una estimación de 15 días que permitió ejecutar 5 historias de usuario incluido la aplicación móvil del lado del usuario y la aplicación del lado del conductor, de ahí que las historias de usuario estimadas son

Aplicación Cliente:

- Notificar llegada del conductor
- Calificar servicio
- Iniciar sesión

Aplicación Taxis:

- Reportar llegada
- Iniciar Carrera
- Terminar Carrera

3.1.3.5 Daily

Esta ceremonia se llevó a cabo de forma presencial, con una duración de 10 a 15 minutos informando de las actividades a realizar durante las siguientes 24 horas.

3.1.3.6 Sprint Review

Esta tipo de reunión se llevó a cabo en una reunión informal al final del sprint informado a los usuarios y taxistas el desarrollo y avance que se llevó en el transcurso de los 15 días que duro cada sprint.

3.1.3.7 Sprint Retrospective

En este punto todo el Scrum Team se analiza y autoevalúa como ha sido el trabajo en el transcurso del sprint llegando a tener una retroalimentación. Por consiguiente se obtienen las cosas buenas, malas y por mejorar que quedan como lección aprendida después de cada Sprint.

3.2 IMPLEMENTACIÓN EXTREME PROGRAMMING

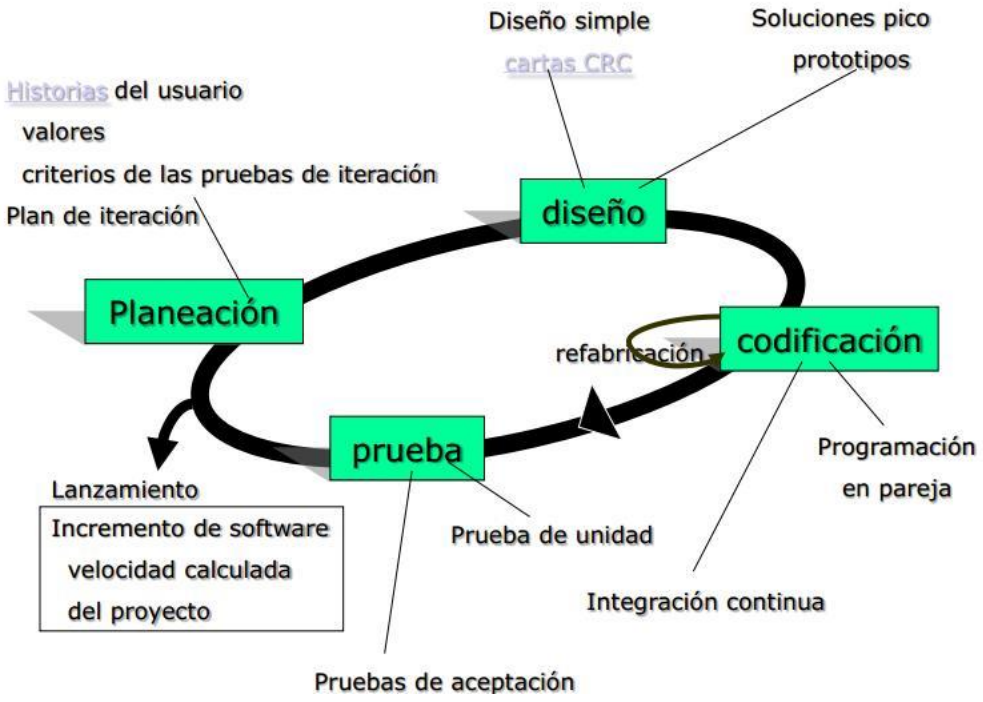


Figura 16. Metodología XP

[52]

3.2.1 Planificación

En la fase de planificación se realizaron reuniones previas con los taxistas y usuarios del servicio de taxi obteniendo información a partir de las historias de usuario acorde a las funcionalidades descritas para las aplicaciones móviles, a partir de esto se priorizaron las historias de usuario es decir el Product Backlog asignando un peso y tamaño por cada historia usando la herramienta planning póker, visualizadas en un tablero de la herramienta *Trello*. Además de aplicar la técnica ágil inception para tener una conceptualización compartida de los productos a realizar resolviendo una serie de interrogantes.

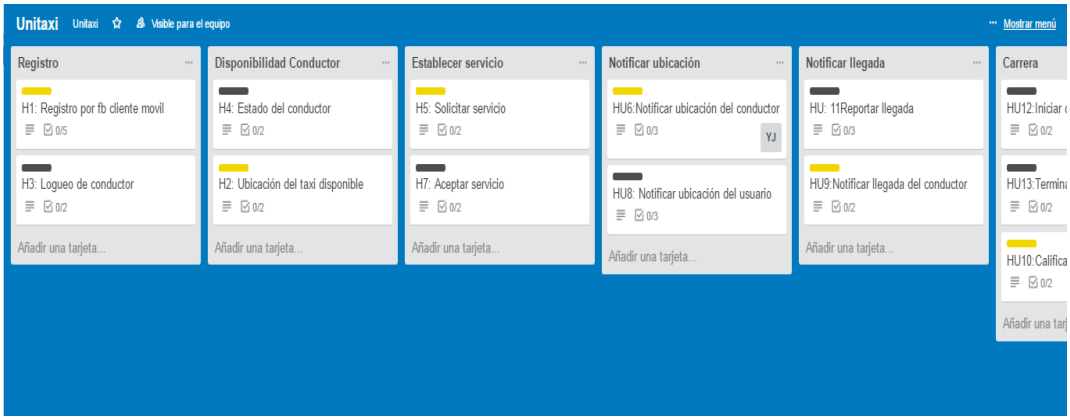


Figura 17. Tablero Kanban [Fuente propia]

3.2.2 Diseño

Acuerdo al estudio realizado por el IDC dice que la plataforma más utilizada por los usuarios es Android con un total del 82.8 % de la población es por ello que en esta fase se escogió dicha plataforma y se definió la estructura del sistema.

Period	Android	iOS	Windows Phone	BlackBerry OS	Others
2015Q2	82.8%	13.9%	2.6%	0.3%	0.4%
2014Q2	84.8%	11.6%	2.5%	0.5%	0.7%
2013Q2	79.8%	12.9%	3.4%	2.8%	1.2%
2012Q2	69.3%	16.6%	3.1%	4.9%	6.1%

Source: IDC, Aug 2015

Figura 18. Estudio IDC

[17]

Gracias a la selección de dicha plataforma se pudo crear un diseño simple y sencillo, esto acelero el tiempo y esfuerzo de desarrollo usando las herramientas Android Studio, material design, *retrofit*, *RXJava* y *Realm.*

3.2.3 Desarrollo

Las funcionalidades de las aplicaciones móviles se desarrollaron en la herramienta *Android Studio* mediante clases y métodos codificados en Java y archivos XML cumpliendo con las funcionalidades previamente expresadas por los usuarios, debido a las especificaciones se vio en la obligación de utilizar repositorios y librerías desarrolladas por terceros que facilitó la codificación de dichas funcionalidades del sistema, utilizando un sistema de control de versiones llamado Bitbucket que es un servidor de control de versiones Git.

3.2.4 Pruebas

Se realizaron las pruebas unitarias para la detección de errores y test del correcto funcionamiento de las historias de usuario desarrolladas cumpliendo las especificaciones de los usuarios y taxistas, estas pruebas se realizaron con el equipo de desarrollo y los usuarios finales otorgando la aprobación del correcto desarrollo de cada una de las funcionalidades.

4. INGENIERÍA DEL PROYECTO

4.1 IMPLEMENTACIÓN SPRINT 0.

Como parte de la estrategia de implementación del Sprint 0, además de las actividades típicas del Agile Inception, también se trabajo la diagramación de la base de datos, diagramas de clase y una arquitectura como insumos para el Sprint uno (1).

4.1.1 Diagrama de Bases de Datos

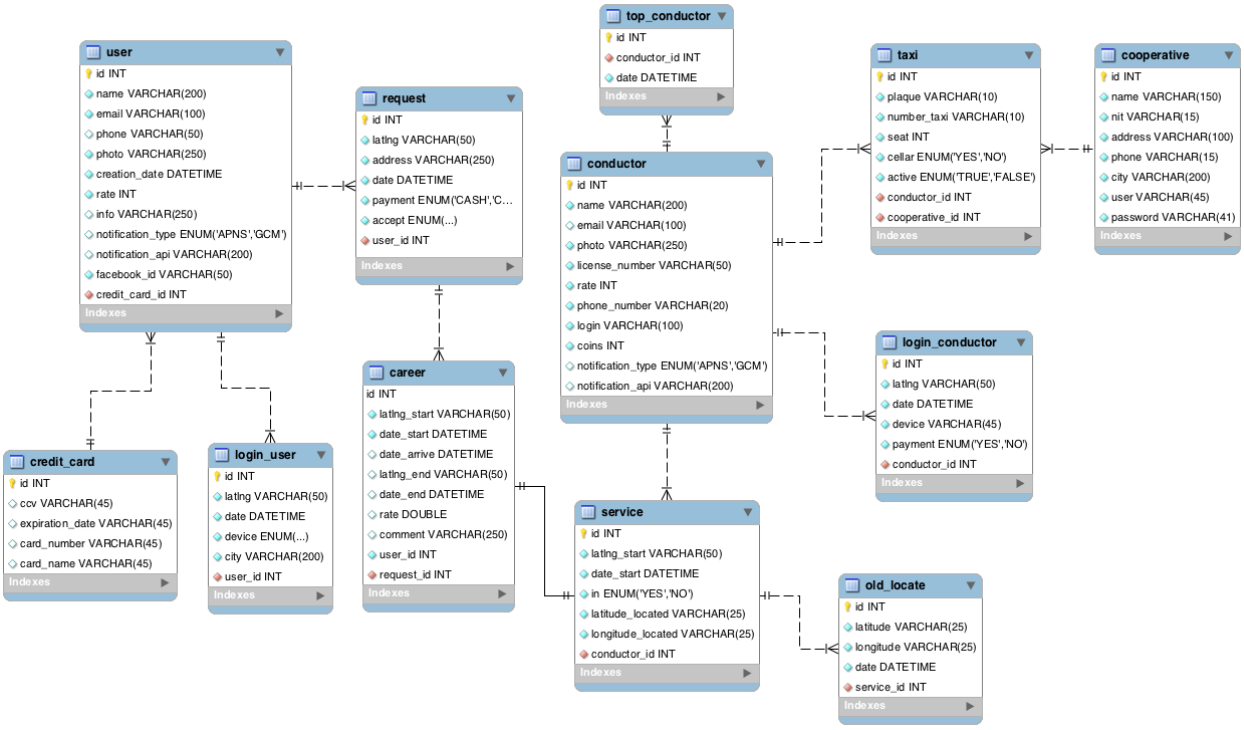


Figura 19. Diagrama de Bases de Datos [Fuente propia]

Las tablas peticionadas mediante los aplicativos son: User, request, conductor, taxi, login_conductor, login_user.

Login_User: A esta tabla se accede cada momento que el usuario inicia en la aplicación.

Login_conductor: En esta tabla se accede cuando el conductor inicia en la aplicación.

User: De la tabla usuario se obtienen los datos para mostrar al conductor cuando este acepta el servicio como lo son nombre, teléfono, foto y ubicación. La ubicación es tomada de la tabla request.

Conductor: De esta tabla se obtienen los datos para mostrar a los usuarios cuando dicho conductor acepta el servicio como lo es nombre foto teléfono y placa del vehículo que se obtiene de la tabla taxi.

4.1.2 Diagrama de Clases

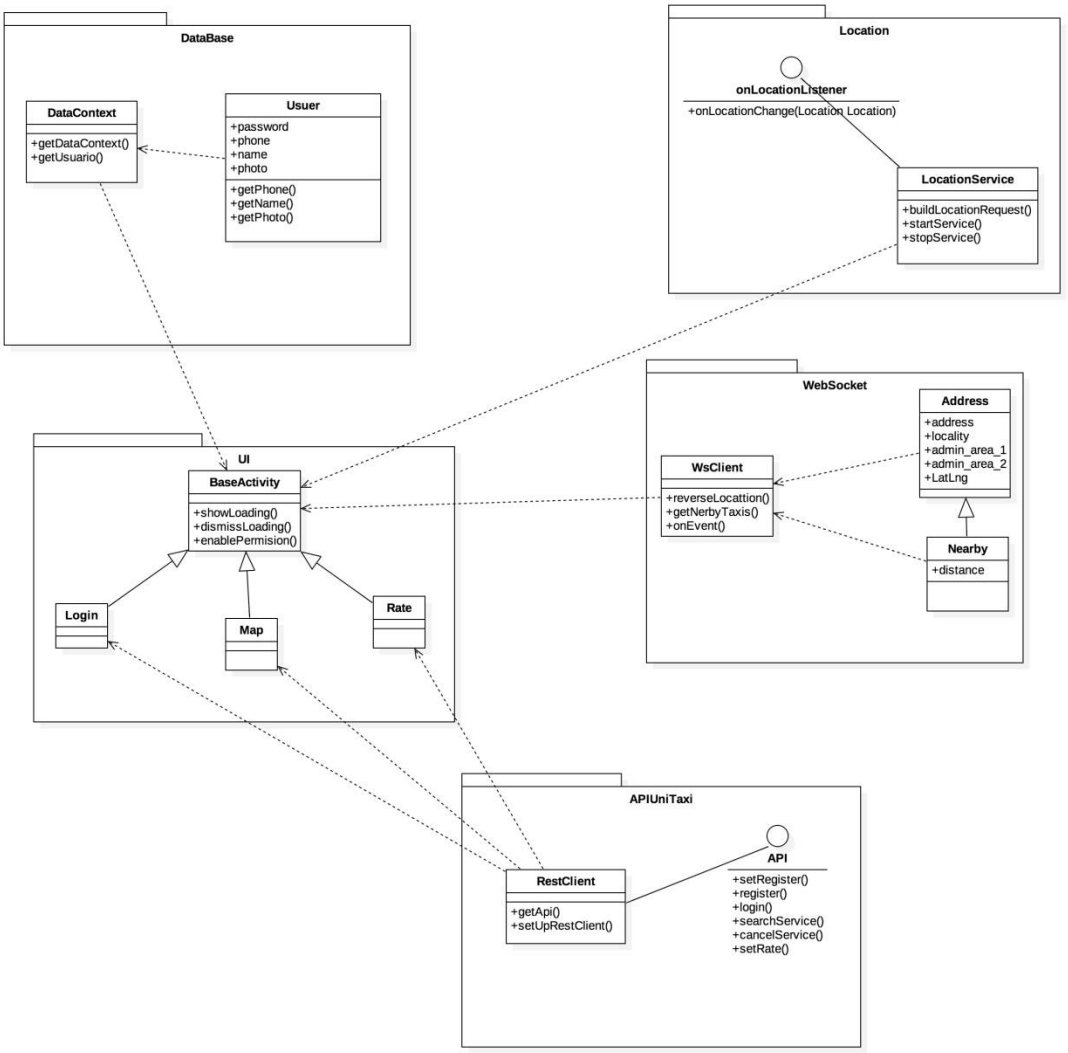


Figura 20. Diagrama de clases [Fuente Propia]

4.1.3 Arquitectura del sistema

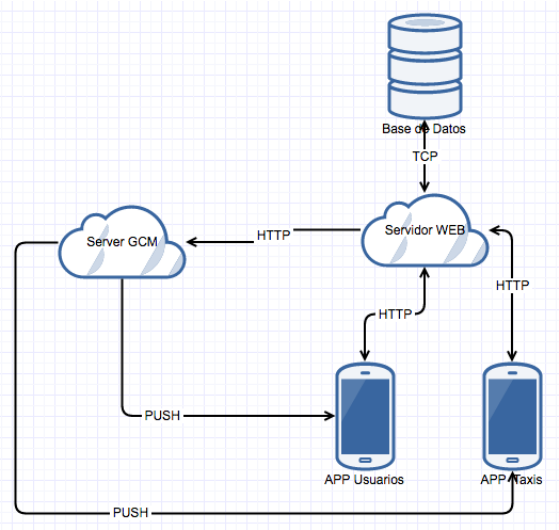


Figura 21. Arquitectura del sistema [Fuente Propia]

Los clientes móviles mediante peticiones HTTP envían al servidor los datos ingresados por los usuarios o conductores, estos son procesados para atender las necesidades de los clientes con respuestas a través del mismo canal HTTP establecido por el cliente. Cuando la comunicación lo requiere esta debe ser cliente-servidor-cliente arquitectura no soportada por los servidores tradicionales, por esto se desvía a un tercero (GCM) para que cumpla esa necesidad, para comportarse cliente - servidor - GCM - cliente, en la entrega de información de un usuario a un conductor o viceversa (Chat Conversation End).

4.1.4 Implementación Inception

Al realizar el incepción participaron todas las personas directamente implicadas en el desarrollo de las aplicaciones móviles con el fin de tener una visión compartida

y aclarar las dudas que se tienen acerca del producto a realizar, así pues se deben resolver las siguientes preguntas:

4.1.4.1 ¿Porque se está aquí? ¿Porque se está haciendo este producto?

Proporcionar una solución software con el desarrollo de dos aplicaciones móviles que facilita la solicitud del servicio de taxi en la ciudad de Popayán, de ahí que la primera aplicación llamada “UniTaxi Manager” del lado del taxista le permite a este aceptar los servicios solicitados por parte del usuario, obtener información personal y visualizar la ubicación del usuario que solicito el servicio. Además la segunda aplicación móvil “UniTaxi” del lado del usuario permite solicitar el servicio de taxi, visualizar la información del taxista que acepto el servicio y visualizar la ubicación constaste de éste. Estas aplicaciones móviles permiten la gestión del servicio de taxi.

4.1.4.2 Elevator Pitch- ¿Cómo defender el proyecto, si se tiene un tiempo determinado?

Aplicaciones móviles que facilitan la solicitud del servicio de taxi en la ciudad de Popayán, una aplicación del lado del taxista quien acepta el servicio y una del lado del usuario quien solicita el servicio, además de proporcionar una seguridad al conocer quién es el pasajero y conductor.

4.1.4.3 Not List

IN:

- Switch del estado del taxista (Activo - Inactivo)
- Visualizar la ubicación del taxista cada 5 segundos
- Ver la información del usuario
- Ver la información del taxista
- Visualizar en el mapa de *google maps* los taxistas que están en un área determinada.
- Solicitar un servicio de taxi

- El usuario califique el servicio prestado por el taxista

OUT:

- Pasarela de pagos.

DISCUTIR:

- Rutas con mayor frecuencia
- Zonas con mayor usuarios posibles

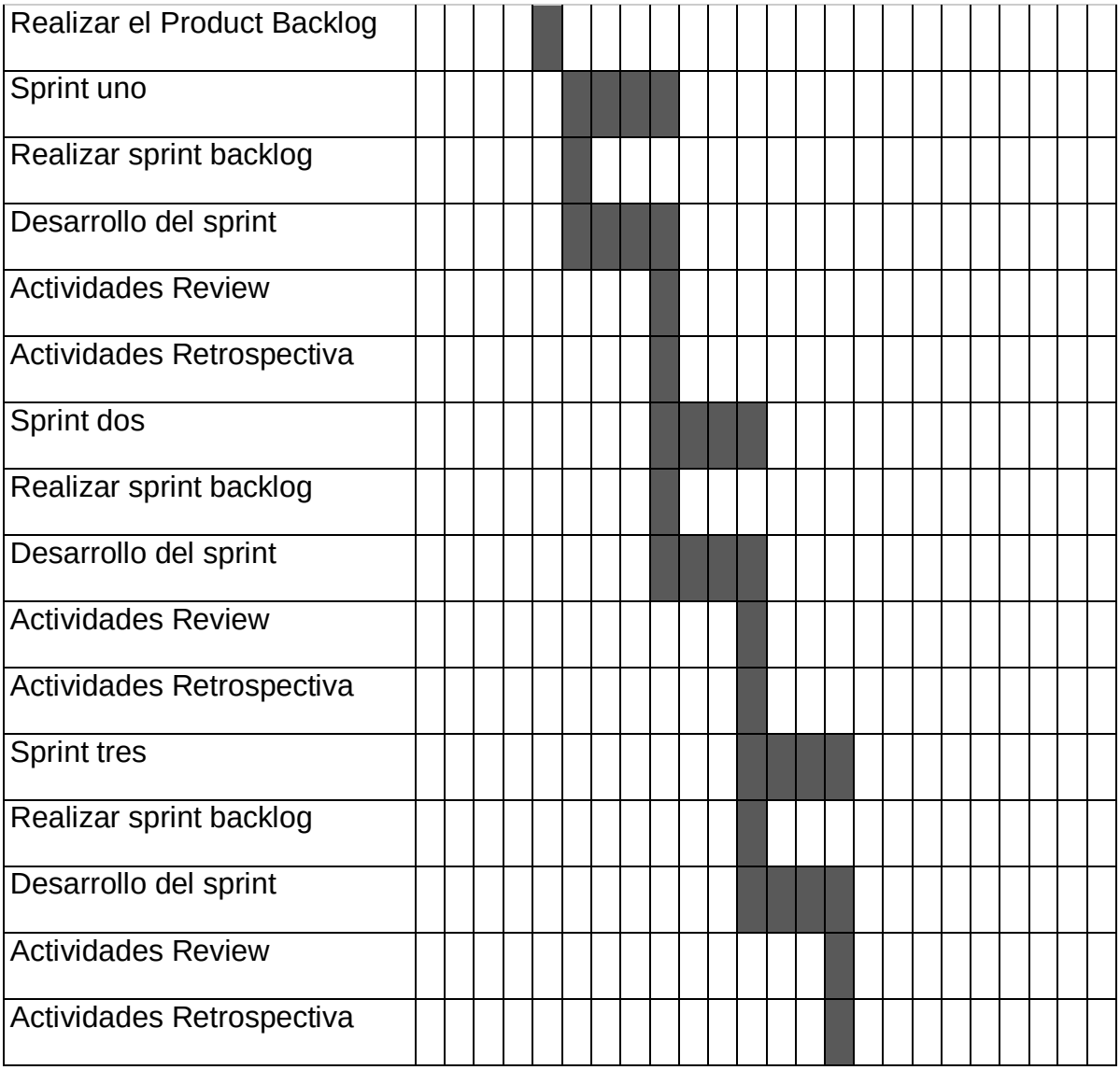
4.1.4.4 Que te quita el sueño.

- ¿Cuáles son tus miedos?
  - Las empresas de taxis no utilicen la aplicación.
  - La comunidad Payanesa no utilice la aplicación para solicitar un servicio.
- ¿Cómo evitarlo?
  - Socializar la aplicación móvil con las empresas de taxi.
  - Garantizar seguridad al usuario final con el uso de la aplicación.
- ¿Cómo minimizar el riesgo?
  - Establecer canales de comunicación.
- ¿Qué nos mantiene despiertos en las noches?
  - Riesgo 1: El usuario no quede a gusto con la aplicación desarrollada.
  - Riesgo 2: Aplicaciones móviles no sean intuitivas.
  - Impedimento 1: Los recursos (herramientas) no sean suficientes para el equipo de trabajo.
  - Impedimento 2: La estimación del tiempo de desarrollo no se ajuste al cronograma establecido.

4.1.4.5 Calcula un tamaño

Tabla 1. Cronograma del Proyecto

CRONOGRAMA DEL PROYECTO Desarrollo de aplicaciones móviles que optimice los tiempos de respuesta en las solicitudes de los servicios de taxi en la ciudad de Popayán Cauca, Colombia.																								
ACTIVIDADES	**SEMANAS**																							
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Elección del Tema e Información																								
Formulación del Problema																								
Revisiones Bibliográficas																								
Búsqueda de referencias																								
Estado del Arte o Marco Referencial																								
Marco Teórico																								
Selección de Metodología																								
Aplicación de Análisis del Resultado																								
Elaboración del Informe																								
Sustentación del Trabajo																								
Publicación del Trabajo																								
Desarrollo Aplicación - Mediante Scrum y XP																								
Sprint cero																								



4.1.4.6 Cuáles son las prioridades

- Cumplir con el cronograma de desarrollo.
- Las herramientas utilizadas se ajuste a lo necesitado.
- Contar con los recursos necesarios.
- Establecer una comunicación entre el equipo de desarrollo.

4.1.4.7 Sprint Review

Con las herramientas usadas como *agile inception* en el *sprint* 0, se obtuvo una visión completa de los productos software a desarrollar, estableciendo las funcionalidades a implementar definiendo el cronograma de actividades a lo largo del proyecto.

4.1.4.8 Retrospectiva

En el *sprint* 0 se definieron por parte del equipo de desarrollo las tecnologías a usar para la implementación de las aplicaciones móviles en el sistema operativo Android, además de las tecnologías para la conexión con el Backend.

4.2 Implementación Sprint 1

Para la implementación de este sprint se aplicaron las fases de XP con una estimación de 15 días ejecutando 4 historias de usuario incluyendo la aplicación móvil del lado del usuario y la aplicación del lado del conductor.

Para este sprint se priorizaron las siguientes historias de usuario acorde a las 2 aplicaciones desarrolladas para este proyecto:

Aplicación Cliente:

- Registro de Usuario.
- Ubicación del taxi disponible.

Aplicación Taxis:

- Inicio sesión de conductor.
- Estado del conductor.

4.2.1 Planeación de Sprint 1

En esta etapa se desarrollaron las historias de usuario, Task Card y tarjetas de colaboración del *sprint* 1.

Tabla 2. Historia de usuario registro de Usuario

<table>
<tr><td colspan="2">Historia de usuario</td></tr>
<tr><td colspan="2">Número: HU1</td></tr>
<tr><td colspan="2">Programador Responsable: Team Development</td></tr>
<tr><td colspan="2">Nombre de la historia: Registro de Usuario</td></tr>
<tr><td colspan="2">Nivel de riesgo de desarrollo: Alta</td></tr>
<tr><td colspan="2">Peso de la historia: 20 puntos de la historia</td></tr>
<tr><td colspan="2">Tiempo estimado: 45 Horas</td></tr>
<tr><td colspan="2">Perspectiva del producto
Como: Usuario
Se Requiere: Acceder al sistema
Para: Que los usuarios se puedan registrarse en la aplicación.</td></tr>
<tr><td colspan="2">Criterios de aceptación
El usuario acepte registrarse con Facebook o se registre en la aplicación móvil.
Facebook proporcione la información del usuario.
Enviar la información proporcionada por Facebook al backend.</td></tr>
<tr><td>Actividades</td><td>N° de horas de implementación</td></tr>
<tr><td>Crear interfaz gráfica registro de usuario
Permisos de Facebook.
Capturar información del cliente.
Conexión al backend.
Enviar información al backend</td><td>7
16
9
4
9</td></tr>
</table>

Tabla 3. Tarea de Ingeniería Interfaz inicio sesión

TAREA DE INGENIERÍA	
Número de Tarea: 1	**HU1:** Registro de usuario
Nombre de Tarea: Interfaz gráfica registro de usuario	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 20 puntos de historia
Tiempo de implementación estimado: 7 Horas	
Fecha Inicio: 25/01/2016	**Fecha Fin:** 12/02/2016
Programador Responsable: Team Development	
Descripción: Crear interfaz gráfica para iniciar sesión en la aplicación móvil del cliente a través de Facebook o registro de usuario.	

Tabla 4. Tarea de Ingeniería permisos de Facebook

TAREA DE INGENIERÍA	
Número de Tarea: 2	**HU1:** Registro de Usuario
Nombre de Tarea: Permisos de Facebook	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 20 puntos de historia
Tiempo de implementación estimado: 16 Horas	
Fecha Inicio: 25/01/2016	**Fecha Fin:** 12/02/2016
Programador Responsable: Team Development	

Descripción: Crear una aplicación en *Facebook developer* solicitando los permisos necesarios para iniciar sesión.

Tabla 5. Tarea de Ingeniería Capturar información del cliente

	TAREA DE INGENIERÍA
Número de Tarea: 3	**HU1:** Registro de Usuario
Nombre de Tarea: Capturar información del cliente	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 20 puntos de historia
Tiempo de implementación estimado: 9 Horas	
Fecha Inicio: 25/01/2016	**Fecha Fin:** 12/02/2016
Programador Responsable: Team Development	
Descripción: Obtener los datos necesarios para el inicio de sesión por Facebook o registro de usuario como lo son datos personales tales como nombre, genero, foto, correo electrónico y clave.	

Tabla 6. Tarea de Ingeniería Conexión al Backend

	TAREA DE INGENIERÍA
Número de Tarea: 4	**HU1:** Registro de Usuario
Nombre de Tarea: Conexión al Backend	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 20 puntos de historia
Tiempo de implementación estimado: 4 Horas	

Fecha Inicio: 25/01/2016	**Fecha Fin:** 12/02/2016
Programador Responsable: Team Development	
Descripción: Se realizan mediante peticiones *HTTP* al servidor mediante la librería *retrofit*.	

Tabla 7. Tarea de Ingeniería Enviar información al Backend

	TAREA DE INGENIERÍA
Número de Tarea: 5	**HU1:** Registro de Usuario
Nombre de Tarea: Enviar información al Backend	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 20 puntos de historia
Tiempo de implementación estimado: 9 Horas	
Fecha Inicio: 25/01/2016	**Fecha Fin:** 12/02/2016
Programador Responsable: Team Development	
Descripción: Consumo e integración de servicios web por medio de peticiones *HTTP.*	

Tabla 8. Tarjeta colaboración HU1 Registro de Usuario

Usuario	
Responsabilidades	**Colaboradores**
• +getPhone() • +getName() • +getPhoto()	• DataContex • BaseActivit • Login

Tabla 9. Historia de usuario Ubicación del taxi disponible

Historia de usuario	
Número: HU2	
Programador Responsable: Team Development	
Nombre de la historia: Ubicación del taxi disponible.	
Nivel de riesgo de desarrollo: Alta	
Peso de la historia: 8 puntos de la historia	
Tiempo estimado: 25 Horas	
Perspectiva del producto **Como:** Usuario **Se Requiere:** Conocer en un radio de 5 km los taxis disponibles **Para:** Solicitar un servicio de taxi	
Criterios de aceptación • Visualizar todos los taxis disponibles en un radio de 5 km donde se encuentra ubicado el usuario. • Actualizar la ubicación del taxi cada 5 segundos.	
Actividades	**N° de horas de implementación**
• Interfaz grafica • Radio de taxis disponible	**10** **15**

Tabla 10. Tarea de Ingeniería Interfaz grafica

TAREA DE INGENIERÍA	
Número de Tarea: 1	**HU2:** Ubicación del taxi disponible
Nombre de Tarea: Interfaz gráfica	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 8 puntos de historia
Tiempo de implementación estimado: 10 Horas	

Fecha Inicio: 25/01/2016	**Fecha Fin:** 12/02/2016
Programador Responsable: Team Development	
Descripción: Crear interfaz gráfica para visualizar la ubicación de los taxis disponibles en un radio de 5 km.·	

Tabla 11. Tarea de Ingeniería Radio de taxis disponibles

	TAREA DE INGENIERÍA
Número de Tarea: 2	**HU2:** Ubicación del taxi disponible
Nombre de Tarea: Radio de taxis disponibles	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 8 puntos de historia
Tiempo de implementación estimado: 15 Horas	
Fecha Inicio: 25/01/2016	**Fecha Fin:** 12/02/2016
Programador Responsable: Team Development	
Descripción: Mediante la ubicación de la latitud y longitud del cliente se visualizaran los taxis que están cerca de dicha ubicación en el rango de 5km.	

Tabla 12. Tarjeta de colaboración HU2 Ubicación taxi

Usuario	
Responsabilidades	**Colaboradores**
• +getPhone() • +getName() • +getPhoto()	• DataContex • BaseActivit • Login

Tabla 13. Historia de usuario Logueo del conductor

Historia de usuario	
Número: HU3	
Programador Responsable: Team Development	
Nombre de la historia: Inicio de sesión del conductor.	
Nivel de riesgo de desarrollo: Alta	
Peso de la historia: 8 puntos de la historia	
Tiempo estimado: 25 Horas	
Perspectiva del producto **Como:** Conductor **Se Requiere:** Acceder al sistema **Para:** Que los usuarios taxistas se puedan loguear en la aplicación móvil a través de la contraseña generada al registrarse el taxista en el servidor	
Criterios de aceptación • Capturar y enviar la contraseña ingresada al servidor	
Actividades	**N° de horas de implementación**
• Interfaz grafica • Validar contraseña asignada	**10** **15**

Tabla 14. Tarea de Ingeniería interfaz grafica

TAREA DE INGENIERÍA	
Número de Tarea: 1	**HU3:** Inicio de sesión del conductor
Nombre de Tarea: Interfaz grafica	

Tipo de Tarea: Desarrollo	**Puntos estimados:** 8 puntos de historia
Tiempo de implementación estimado: 10 Horas	
Fecha Inicio: 25/01/2016	**Fecha Fin:** 12/02/2016
Programador Responsable: Team Development	
Descripción: Crear interfaz gráfica para el logue de conductor.	

Tabla 15. Tarea de Ingeniería Validar contraseña asignada

	TAREA DE INGENIERÍA
Número de Tarea: 2	**HU3:** Inicio de sesión del conductor
Nombre de Tarea: Validar contraseña asignada	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 8 puntos de historia
Tiempo de implementación estimado: 15 Horas	
Fecha Inicio: 25/01/2016	**Fecha Fin:** 12/02/2016
Programador Responsable: Team Development	
Descripción: Recibir respuesta del Backend con información del taxista.	

Tabla 16. Tarjeta de colaboración HU3 Inicio de sesión del conductor

Usuario	
Responsabilidades	**Colaboradores**
• +getPhone() • +getName() • +getPhoto()	• DataContex • BaseActivit • Login

Tabla 17. Historia de Usuario Estado del conductor

Historia de usuario	
Número: HU4	
Programador Responsable: Team Development	
Nombre de la historia: Estado del conductor	
Nivel de riesgo de desarrollo: Alta	
Peso de la historia: 8 puntos de la historia	
Tiempo estimado: 25 Horas	
Perspectiva del producto **Como:** Conductor **Se Requiere:** Activar o desactivar **Para:** Que los usuarios taxistas tengan un estado activo o inactivo para la disponibilidad del servicio como conductor	
Criterios de aceptación • El conductor pueda cambiar de estado activo o inactivo	
Actividades	**N° de horas de implementación**
• Interfaz grafica • Establecer estado	**10** **15**

Tabla 18. Tarea de ingeniería Interfaz Grafica

TAREA DE INGENIERÍA	
Número de Tarea: 1	**HU4:** Estado del conductor
Nombre de Tarea: Interfaz grafica	

Tipo de Tarea: Desarrollo	**Puntos estimados:** 8 puntos de historia
Tiempo de implementación estimado: 10 Horas	
Fecha Inicio: 25/01/2016	**Fecha Fin:** 12/02/2016
Programador Responsable: Team Development	
Descripción: Crear interfaz gráfica para el estado del conductor.	

Tabla 19. Tarea de ingeniería Establecer estado

	TAREA DE INGENIERÍA
Número de Tarea: 2	**HU4** estado del conductor
Nombre de Tarea: Establecer estado	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 8 puntos de historia
Tiempo de implementación estimado: 15 Horas	
Fecha Inicio: 25/01/2016	**Fecha Fin:** 12/02/2016
Programador Responsable: Team Development	
Descripción: Establecer un estado de activo o inactivo para el conductor pueda prestar el servicio.	

4.2.2 Ejecución del Sprint 1

4.2.2.1 Diseño

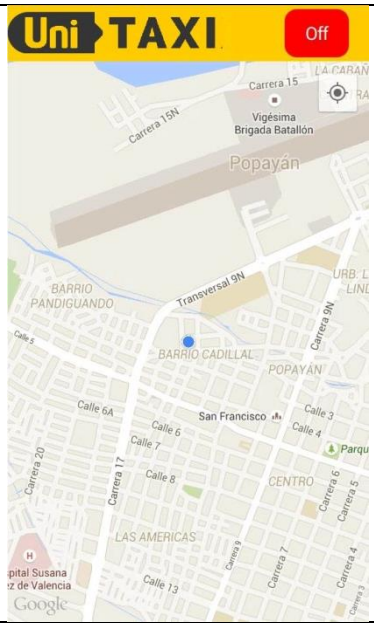
Figura 22. Interfaz Iniciar Sesión del conductor [Fuente propia]

Interfaz gráfica de inicio de Sesión de la aplicación para el conductor poder iniciar sesión con la contraseña previamente asignada por parte del Backend.

4.2.2.2 Desarrollo

Tabla 20. Evento programado Logueo conductor

Evento Programado inicio sesión del conductor
Resultados del inicio de sesión



Eventos Programados

```
}

@Override
public void onEvent(Location location) {
    super.onEvent(location);
    myLocation=location;
    if (BdUtils.getApplicationDataContext(this).getUsuariosSet().getMyUser()!=null) {
        ui.showDialog("Loading");
        if (myLocation!= null)
        {
            loginbd(myLocation);
        }

    }
    //ui.dissmisDialgog();



}

private void loginbd(Location myLocation) {
    //ui.showDialog(getResources().getString(R.string.loading));
    cont++;
    if (cont == limit)
    {
        if(BdUtils.getApplicationDataContext(this).getUsuariosSet().getMyUser()!=null) {

            if(myLocation!=null) {
                Gcmi_id = BdUtils.getApplicationDataContext(this).getUsuariosSet().getMyUser().getGmc();
                Map<String, String> param = RestClient.getParamsLogin(BdUtils.getApplicationDataContext(this)
                    .getUsuariosSet().getMyUser().getPass(), myLocation.getLatitude() + "," + myLocation.getLongitude(), Gcmi_id);
                RestClient.get().login(param, this);

            }


        }
    }
}
```

4.2.2.3 Pruebas

Tabla 21. Prueba funcional Inicio de sesión del conductor

Historia de	Inicio de sesión del conductor

usuario	
Propósito	Iniciar sesión en el aplicación móvil
Requisito	Tener contraseña asignada
Pasos	Se ingresa a la aplicación móvil Unitaxi manager y se ingresa la clave previamente asignada en el campo contraseña y presiona el botón loguear.
Resultados esperados	Visualizar el mapa de google con la ubicación actual del conductor y la opción de activar o desactivar estado.

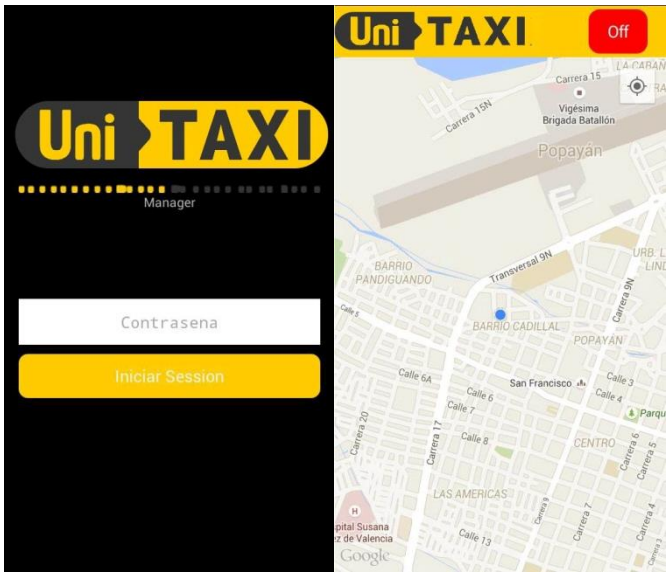


Figura 23. Interfaces graficas de inicio de sesión [Fuente propia]

4.2.3 Sprint Review 1

Al finalizar el sprint review el equipo de desarrollo entrega todas las funcionalidades del sprint 1 que son las siguientes:

- Registro de usuario cliente móvil.
- Ubicación del taxi disponible.

- Inicio de sesión del conductor
- Estado del conductor.

4.2.4 Sprint Retrospective 1

Tabla 22. Retrospectiva Sprint 1

 • La documentación oficial de Android nos facilitó en el desarrollo del aplicativo. • La experiencia en el desarrollo de aplicaciones móviles.	 • Tiempo de aprendizaje. • Cambio de tecnología HTTP request.
 • Validación inicio de sesión por Facebook. • Switch activar o desactivar conductor.	 • Agradecimientos al Scrum master por verificar el cumplimiento de la metodología Scrum.

4.3 Implementación Sprint 2

Para la implementación de este sprint se aplicaron las fases de XP con una estimación de 15 días ejecutando 4 historias de usuario incluyendo la aplicación móvil del lado del usuario y la aplicación del lado del conductor

Para este sprint se priorizaron las siguientes historias de usuario acorde a las 2 aplicaciones desarrolladas para este proyecto:

Aplicación Cliente:

- Solicitar servicio.
- Notificar Ubicación del conductor.

Aplicación Taxis:

- Aceptar servicio.
- Notificar ubicación del usuario

4.3.1 Planeación del Sprint 2

En esta etapa se desarrollaron las historias de usuario, Task Card y tarjetas de colaboración del sprint 2.

Tabla 23. Historia de Usuario Solicitar Servicio

Historia de usuario
Número: HU5
Programador Responsable: Team Development
Nombre de la historia: Solicitar servicio.
Nivel de riesgo de desarrollo: Alta
Peso de la historia: 13 puntos de la historia
Tiempo estimado: 25 Horas
Perspectiva del producto **Como:** Usuario **Se Requiere:** Solicitar un servicio **Para:** Que los usuarios puedan solicitar un servicio en la ubicación donde se encuentran

Criterios de aceptación	
• Visualizar las ubicaciones de los taxis que estas en el área exacta de ubicación del usuario • Proporcionar mapa mediante *google maps*. • Enviar la ubicación y solicitud al Backend.	
Actividades	**N° de horas de implementación**
• Crear interfaz gráfica. • Solicitar servicio de taxi • Cancelar servicio	**10** **8** **7**

Tabla 24. Tarea de ingeniería Interfaz Grafica

TAREA DE INGENIERÍA	
Número de Tarea: 1	**HU5:** Solicitar servicio
Nombre de Tarea: Interfaz gráfica	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 13 puntos de historia
Tiempo de implementación estimado: 10 Horas	
Fecha Inicio: 15/02/2016	**Fecha Fin:** 04/03/2016
Programador Responsable: Team Development	
Descripción: Crear interfaz gráfica para que el usuario solicite el servicio de taxi acorde a la ubicación de esté.	

Tabla 25. Tarea de ingeniería Solicitar servicio de taxi

TAREA DE INGENIERÍA	
Número de Tarea: 2	**HU5:** Solicitar servicio

Nombre de Tarea: Solicitar servicio de taxi	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 13 puntos de historia
Tiempo de implementación estimado: 8 Horas	
Fecha Inicio: 15/02/2016	**Fecha Fin:** 04/03/2016
Programador Responsable: Team Development	
Descripción: Enviar ubicación proporcionada por *google maps* y datos del usuario al servidor solicitando un servicio de taxi.	

Tabla 26. Tarea de ingeniería Cancelar servicio

	TAREA DE INGENIERÍA
Número de Tarea: 3	**HU5:** Solicitar servicio
Nombre de Tarea: Cancelar servicio de taxi	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 7 puntos de historia
Tiempo de implementación estimado: 8 Horas	
Fecha Inicio: 15/02/2016	**Fecha Fin:** 04/03/2016
Programador Responsable: Team Development	
Descripción: Cancelar el servicio previamente solicitado por parte del usuario.	

Tabla 27. Tarjeta colaboración HU5 Solicitar servicio

Usuario	
Responsabilidades	**Colaboradores**

• +getPhone() • +getName() • +getPhoto()	• DataContex • BaseActivit • Login

Tabla 28. Historia de Usuario Notificar ubicación del conductor

Historia de usuario	
Número: HU6	
Programador Responsable: Team Development	
Nombre de la historia: Notificar ubicación del conductor.	
Nivel de riesgo de desarrollo: Alta	
Peso de la historia: 20 puntos de la historia	
Tiempo estimado: 35 Horas	
Perspectiva del producto **Como:** Usuario **Se Requiere:** Conocer la ubicación del taxi más cercano que acepto el servicio **Para:** Saber la ubicación y proximidad en la que se encuentra el taxi cada 5 segundos	
Criterios de aceptación • Visualizar ubicación del taxi en el mapa. • Actualizar la ubicación del taxi cada 5 segundos.	
Actividades	**N° de horas de implementación**
• Interfaz grafica • Recibir ubicaciones constantes del conductor • Visualizar ubicación del conductor	**7** **17** **11**

Tabla 29. Tarea de ingeniería Interfaz grafica

TAREA DE INGENIERÍA	
Número de Tarea: 1	**HU6:** Notificar ubicación del conductor
Nombre de Tarea: Interfaz gráfica	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 20 puntos de historia
Tiempo de implementación estimado: 7 Horas	
Fecha Inicio: 15/02/2016	**Fecha Fin:** 04/03/2016
Programador Responsable: Team Development	
Descripción: Crear interfaz gráfica para visualizar la ubicación del taxi que acepto el servicio en el mapa.	

Tabla 30. Tarea de ingeniería Recibir ubicaciones del conductor

TAREA DE INGENIERÍA	
Número de Tarea: 2	**HU6:** Notificar ubicación del conductor
Nombre de Tarea: Recibir ubicaciones constantes del conductor	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 20 puntos de historia
Tiempo de implementación estimado: 17 Horas	
Fecha Inicio: 15/02/2016	**Fecha Fin:** 04/03/2016
Programador Responsable: Team Development	
Descripción: Capturar la latitud y longitud que envía el servidor cada 5 segundos mediante HTTP.	

Tabla 31. Tarea de ingeniería Visualizar ubicación del conductor

TAREA DE INGENIERÍA	
Número de Tarea: 3	**HU6:** Notificar ubicación del conductor
Nombre de Tarea: Visualizar ubicación del conductor	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 20 puntos de historia
Tiempo de implementación estimado: 11 Horas	
Fecha Inicio: 15/02/2016	**Fecha Fin:** 04/03/2016
Programador Responsable: Team Development	
Descripción: Visualizar la ubicación actualizada del taxi en el mapa.	

Tabla 32. Tarjetas de colaboración HU6 Notificar ubicación del conductor

Usuario	
Responsabilidades	**Colaboradores**
• +getPhone() • +getName() • +getPhoto()	• DataContex • BaseActivit • Login

Tabla 33. Historia de Usuario Aceptar Servicio

Historia de usuario
Número: HU7
Programador Responsable: Team Development

Nombre de la historia: Aceptar servicio	
Nivel de riesgo de desarrollo: Alta	
Peso de la historia: 13 puntos de la historia	
Tiempo estimado: 25 Horas	
Perspectiva del producto **Como:** Conductor **Se Requiere:** Aceptar o ignorar un servicio **Para:** Que los usuarios taxistas puedan aceptar servicios disponibles en la aplicación móvil a través de notificaciones de servicios disponibles enviados por el servidor.	
Criterios de aceptación • Capturar y renderizar la información del servicio.	
Actividades	**N° de horas de implementación**
• Interfaz grafica • Aceptar servicio de taxi	**10** **15**

Tabla 34. Tarea de ingeniería Interfaz gráfica

	TAREA DE INGENIERÍA
Número de Tarea: 1	**HU7:** Aceptar servicio
Nombre de Tarea: Interfaz gráfica	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 13 puntos de historia
Tiempo de implementación estimado: 10 Horas	
Fecha Inicio: 15/02/2016	**Fecha Fin:** 04/03/2016
Programador Responsable: Team Development	

Descripción: Crear interfaz gráfica para que los taxis puedan aceptar servicios disponibles.

Tabla 35. Tarea de ingeniería Aceptar servicio

TAREA DE INGENIERÍA	
Número de Tarea: 2	**HU7:** Aceptar servicio
Nombre de Tarea: Aceptar servicio	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 13 puntos de historia
Tiempo de implementación estimado: 15 Horas	
Fecha Inicio: 15/02/2016	**Fecha Fin:** 04/03/2016
Programador Responsable: Team Development	
Descripción: Enviar la información del taxi y del servicio que fue aceptado al servidor.	

Tabla 36. Tarjeta de colaboración HU7 Aceptar servicio

Usuario	
Responsabilidades	**Colaboradores**
• +getPhone() • +getName() • +getPhoto()	• DataContex • BaseActivit • Login

Tabla 37. Historia de Usuario Notificar ubicación del usuario

Historia de usuario	
Número: HU8	
Programador Responsable: Team Development	
Nombre de la historia: Notificar ubicación del usuario	
Nivel de riesgo de desarrollo: Alta	
Peso de la historia: 20 puntos de la historia	
Tiempo estimado: 35 Horas	
Perspectiva del producto **Como:** Conductor **Se Requiere:** Ver ubicación del pasajero. **Para:** Que los usuarios taxistas puedan ver la ubicación del pasajero y puedan dirigirse a su ubicación.	
Criterios de aceptación • Ver ubicación del servicio en el mapa. • Ver la información del pasajero.	
Actividades	**N° de horas de implementación**
• Interfaz grafica • Recibir ubicación del usuario • Visualizar ubicación del usuario	**7** **17** **11**

Tabla 38. Tarea de Ingeniería Interfaz gráfica

TAREA DE INGENIERÍA	
Número de Tarea: 1	**HU8:** Notificar ubicación del usuario
Nombre de Tarea: Interfaz gráfica	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 20 puntos de historia

Tiempo de implementación estimado: 7 Horas	
Fecha Inicio: 15/02/2016	**Fecha Fin:** 04/03/2016
Programador Responsable: Team Development	
Descripción: Crear interfaz gráfica para ver la ubicación y datos del usuario	

Tabla 39. Tarea de Ingeniería Recibir ubicación del usuario

	TAREA DE INGENIERÍA
Número de Tarea: 2	**HU8:** Notificar ubicación del usuario
Nombre de Tarea: Recibir ubicación del usuario	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 20 puntos de historia
Tiempo de implementación estimado: 17 Horas	
Fecha Inicio: 15/02/2016	**Fecha Fin:** 04/03/2016
Programador Responsable: Team Development	
Descripción: Capturar la información del servicio y pasajero enviada por el servidor.	

Tabla 40. Tarea de ingeniería Visualizar ubicación del usuario

	TAREA DE INGENIERÍA
Número de Tarea: 3	**HU8:** Notificar ubicación del usuario
Nombre de Tarea: Visualizar ubicación del usuario	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 20 puntos de historia

Tiempo de implementación estimado: 11 Horas	
Fecha Inicio: 15/02/2016	**Fecha Fin:** 04/03/2016
Programador Responsable: Team Development	
Descripción: Visualizar la información de la ubicación del servicio y la información del pasajero.	

4.3.2 Ejecución del Sprint 2

4.3.2.1 Diseño

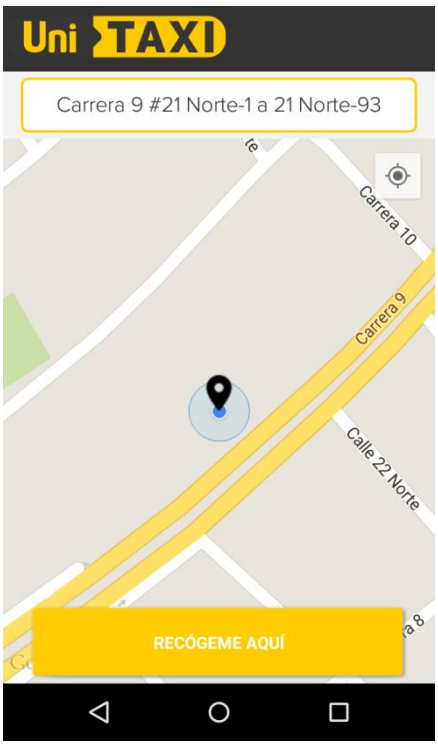
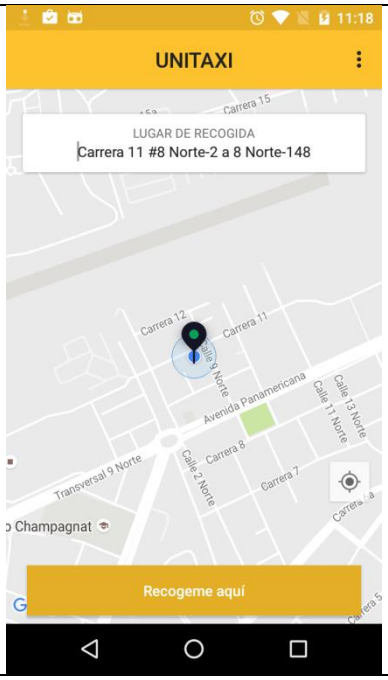


Figura 24. Interfaz solicitar servicio de taxi [Fuente propia]

Interfaz gráfica solicitar servicio de taxi del lado del usuario, en el que se seleccionará el botón “recógeme aquí” para iniciar la solitud.

4.3.2.2 Desarrollo

Tabla 41. Evento programado Solicitar servicio

Evento Programado Solicitar servicio
Resultados de la solicitud

Eventos Programados

```
}

@Override
public void onEvent(Location location) {
    super.onEvent(location);
    myLocation=location;
    if (BdUtils.getApplicationDataContext(this).getUsuariosSet().getMyUser()!=null) {
        ui.showDialog("Loading");
        if (myLocation!= null)
        {
            loginbd(myLocation);
        }

    }
    //ui.dissmisDialgog();


}

private void loginbd(Location myLocation) {
    //ui.showDialog(getResources().getString(R.string.loading));
    cont++;
    if (cont == limit)
    {
        if(BdUtils.getApplicationDataContext(this).getUsuariosSet().getMyUser()!=null) {

            if(myLocation!=null) {
                Gcmi_id = BdUtils.getApplicationDataContext(this).getUsuariosSet().getMyUser().getGmc();
                Map<String, String> param = RestClient.getParamsLogin(BdUtils.getApplicationDataContext(this)
                    .getUsuariosSet().getMyUser().getPass(), myLocation.getLatitude() + "," + myLocation.getLongitude(), Gcmi_id);
                RestClient.get().login(param, this);

            }


        }
    }

}
```

4.3.2.3 Pruebas

Tabla 42. Prueba funcional Solicitar servicio

Historia de usuario	Solicitar Servicio
Propósito	Usuario solicite el servicio de taxi
Requisito	Iniciar sesión
Pasos	Se ingresa a la aplicación móvil Unitaxi y se selecciona el botón “Recógeme aquí” de la interfaz gráfica con tu longitud y latitud de ubicación visualizada en la interfaz gráfica.
Resultados esperados	Visualizar el mapa de *google* con la ubicación actual del usuario y la opción de solicitar el servicio.

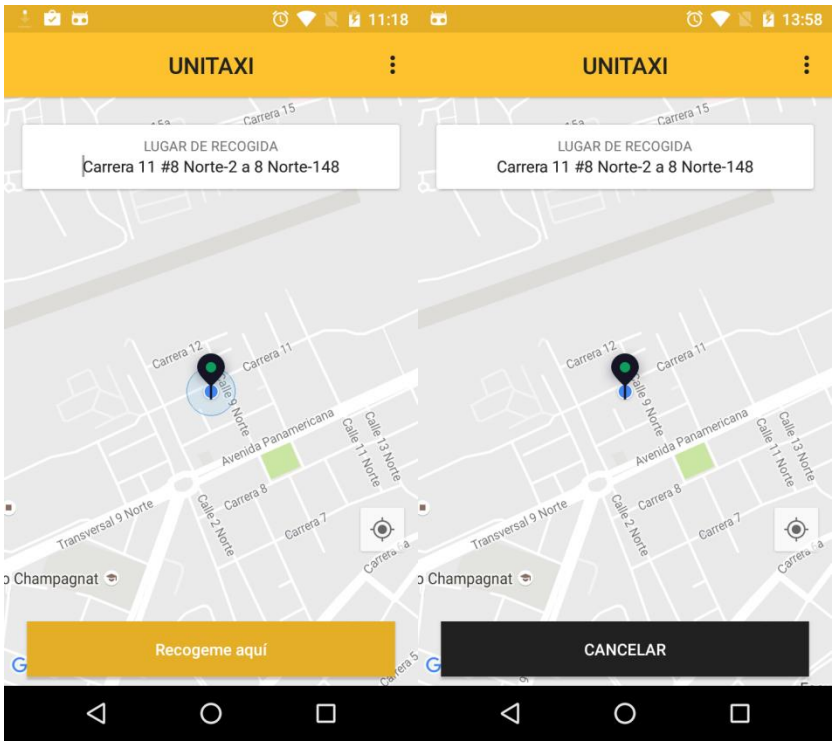


Figura 25. Interfaces graficas Solicitar y cancelar servicio [Fuente propia]

4.3.3 Sprint Review 2.

Al finalizar el sprint review el equipo de desarrollo entrega todas las funcionalidades del sprint 2 que son las siguientes:

- Solicitar servicio.
- Notificar ubicación del conductor.
- Aceptar servicio.
- Notificar ubicación del usuario.

4.3.4 Sprint Retrospective 2.

Tabla 43. Retrospectiva Sprint 2

 • La documentación oficial de Android nos facilitó en el desarrollo del aplicativo. • Uso de *Retrofit* • La experiencia en el desarrollo de aplicaciones móviles.	 • Tiempo de aprendizaje. • Cambio de tecnología HTTP request.
 • Uso de google maps.	 • Agradecimientos al Scrum master por verificar el cumplimiento de la metodología Scrum.

4.4 Implementación Sprint 3

Para la implementación de este sprint se aplicaron las fases de XP con una estimación de 15 días ejecutando 6 historias de usuario incluyendo la aplicación móvil del lado del usuario y la aplicación del lado del conductor.

Para este sprint se priorizaron las siguientes historias de usuario acorde a las 2 aplicaciones desarrolladas para este proyecto:

Aplicación Cliente:

- Notificar llegada del conductor
- Calificar servicio
- Iniciar sesión

Aplicación Taxis:

- Reportar servicio
- Iniciar carrera
- Terminar carrera

4.4.1 Planeación Sprint 3

En esta etapa se desarrollaron las historias de usuario, Task Card y tarjetas de colaboración del sprint 3.

Tabla 44. Historia de Usuario Notificar llegada del conductor

<table>
<tr><td colspan="2">Historia de usuario</td></tr>
<tr><td colspan="2">Número: HU9</td></tr>
<tr><td colspan="2">Programador Responsable: Team Development</td></tr>
<tr><td colspan="2">Nombre de la historia: Notificar llegada del conductor</td></tr>
<tr><td colspan="2">Nivel de riesgo de desarrollo: Alta</td></tr>
<tr><td colspan="2">Peso de la historia: 20 puntos de la historia</td></tr>
<tr><td colspan="2">Tiempo estimado: 17 Horas</td></tr>
<tr><td colspan="2">Perspectiva del producto
Como: Usuario
Se Requiere: Notificar la llegada del taxi solicitado
Para: Que los usuarios reciban una notificación del servicio solicitado</td></tr>
<tr><td colspan="2">Criterios de aceptación
• Notificación de llegada de servicio de taxi.
• Proporcionar mapa mediante google maps.
• Enviar la ubicación y solicitud al backend.</td></tr>
<tr><td>Actividades</td><td>N° de horas de implementación</td></tr>
</table>

• *Push Notifications.* • Visualizar información del conductor	**10** **7**

Tabla 45. Tarea de ingeniería Notificaciones *push*

TAREA DE INGENIERÍA	
Número de Tarea: 1	**HU9:** Notificar llegada del conductor
Nombre de Tarea: *Push Notifications*	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 13 puntos de historia
Tiempo de implementación estimado: 10 Horas	
Fecha Inicio: 07/03/2016	**Fecha Fin:** 25/03/2016
Programador Responsable: Team Development	
Descripción: Notificar a través de google cloud messaging en la aplicación móvil, la llegada del taxi.	

Tabla 46. Tarea de ingeniería Visualizar información del conductor

TAREA DE INGENIERÍA	
Número de Tarea: 2	**HU9:** Notificar llegada del conductor
Nombre de Tarea: Visualizar información del conductor	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 13 puntos de historia
Tiempo de implementación estimado: 7 Horas	

Fecha Inicio: 07/03/2016	**Fecha Fin:** 25/03/2016
Programador Responsable: Team Development	
Descripción: Visualizar en la aplicación móvil del usuario la información básica del conductor.	

Tabla 47. Tarjeta de colaboración HU9 Notificar llegada del conductor

Usuario	
Responsabilidades	**Colaboradores**
• +getPhone() • +getName() • +getPhoto()	• DataContex • BaseActivit • Login

Tabla 48. Historia de Usuario Calificar servicio

Historia de usuario
Número: HU10
Programador Responsable: Team Development
Nombre de la historia: Calificar servicio
Nivel de riesgo de desarrollo: Alta
Peso de la historia: 13 puntos de la historia
Tiempo estimado: 22 Horas
Perspectiva del producto **Como:** Usuario

Se Requiere: Calificar el servicio prestado por el taxista **Para:** Que la empresa de taxis obtenga observaciones del usuario sobre el servicio de taxi.	
Criterios de aceptación • Usuario califique el servicio de taxi. • Calificación por estrellas • El usuario tenga la opción de escribir un comentario sobre el servicio. • Enviar calificación al backend	
Actividades	**N° de horas de implementación**
• Interfaz gráfica • Calificar servicio	**7** **15**

Tabla 49. Tarea de Ingeniera Interfaz gráfica

TAREA DE INGENIERÍA	
Número de Tarea: 1	**HU10:** Calificar servicio
Nombre de Tarea: Interfaz gráfica	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 13 puntos de historia
Tiempo de implementación estimado: 7 Horas	
Fecha Inicio: 07/03/2016	**Fecha Fin:** 25/03/2016
Programador Responsable: Team Development	
Descripción: Crear interfaz gráfica para que el usuario califique o realice un comentario sobre el servicio.	

Tabla 50. Tarea de ingeniería Calificar servicio

TAREA DE INGENIERÍA	
Número de Tarea: 2	**HU10:** Calificar servicio
Nombre de Tarea: Calificar servicio	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 13 puntos de historia
Tiempo de implementación estimado: 15 Horas	
Fecha Inicio: 07/03/2016	**Fecha Fin:** 25/03/2016
Programador Responsable: Team Development	
Descripción: Enviar respuestas al backed del comentario o calificación realizada por el usuario	

Tabla 51. Tarjeta de colaboración HU10 Calificar servicio

Usuario	
Responsabilidades	**Colaboradores**
• +getPhone() • +getName() • +getPhoto()	• DataContex • BaseActivit • Login

Tabla 52. Historia de Usuario Reportar llegada

Historia de usuario
Número: HU11
Programador Responsable: Team Development
Nombre de la historia: Reportar llegada

Nivel de riesgo de desarrollo: Alta	
Peso de la historia: 20 puntos de la historia	
Tiempo estimado: 27 Horas	
Perspectiva del producto **Como:** Conductor **Se Requiere:** Notificar al usuario la llegada del servicio **Para:** Que el usuario tome el servicio de taxi	
Criterios de aceptación • Conductor notifique la llegada	
Actividades	**N° de horas de implementación**
• Interfaz gráfica • Reportar llegada • Visualizar información del cliente	**9** **9** **9**

Tabla 53. Tarea de ingeniería Interfaz gráfica

	TAREA DE INGENIERÍA
Número de Tarea: 1	**HU11:** Reportar llegada
Nombre de Tarea: Interfaz gráfica	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 20 puntos de historia
Tiempo de implementación estimado: 9 Horas	
Fecha Inicio: 07/03/2016	**Fecha Fin:** 25/03/2016
Programador Responsable: Team Development	
Descripción: Crear interfaz gráfica para que el taxista anuncie la llegada.	

Tabla 54. Tarea de ingeniería Reportar llegada

TAREA DE INGENIERÍA	
Número de Tarea: 2	**HU11:** Reportar llegada
Nombre de Tarea: Reportar llegada	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 20 puntos de historia
Tiempo de implementación estimado: 9 Horas	
Fecha Inicio: 07/03/2016	**Fecha Fin:** 25/03/2016
Programador Responsable: Team Development	
Descripción: Enviar información al backend	

Tabla 55. Tarea de ingeniería Visualizar información del cliente

TAREA DE INGENIERÍA	
Número de Tarea: 3	**HU11:** Reportar llegada
Nombre de Tarea: Visualizar información del cliente	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 20 puntos de historia
Tiempo de implementación estimado: 9 Horas	
Fecha Inicio: 07/03/2016	**Fecha Fin:** 25/03/2016
Programador Responsable: Team Development	
Descripción: Visualizar en la aplicación móvil del conductor la información básica del usuario.	

Tabla 56. Tarjeta de colaboración HU11 Reportar llegada

Usuario	
Responsabilidades	**Colaboradores**
• +getPhone() • +getName() • +getPhoto()	• DataContex • BaseActivit • Login

Tabla 57. Historia de Usuario Iniciar carrera

Historia de usuario	
Número: HU12	
Programador Responsable: Team Development	
Nombre de la historia: Iniciar carrera	
Nivel de riesgo de desarrollo: Alta	
Peso de la historia: 13 puntos de la historia	
Tiempo estimado: 22 Horas	
Perspectiva del producto **Como:** Conductor **Se Requiere:** Seleccionar la opción iniciar carrera **Para:** Dar inicio a la carrera	
Criterios de aceptación • Conductor notifique el inicio de la carrera • Enviar información al Backend	
Actividades	**N° de horas de implementación**
• Interfaz gráfica • Iniciar carrera	**7** **15**

Tabla 58. Tarea de Ingeniería Interfaz gráfica

TAREA DE INGENIERÍA	
Número de Tarea: 1	**HU12:** Iniciar carrera
Nombre de Tarea: Interfaz gráfica	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 13 puntos de historia
Tiempo de implementación estimado: 7 Horas	
Fecha Inicio: 07/03/2016	**Fecha Fin:** 25/03/2016
Programador Responsable: Team Development	
Descripción: Crear interfaz gráfica para que el conductor seleccione la opción iniciar carrera	

Tabla 59. Tarea de Ingeniería Iniciar carrera

TAREA DE INGENIERÍA	
Número de Tarea: 2	**HU12:** Iniciar carrera
Nombre de Tarea: Iniciar carrera	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 13 puntos de historia
Tiempo de implementación estimado: 15 Horas	
Fecha Inicio: 07/03/2016	**Fecha Fin:** 25/03/2016
Programador Responsable: Team Development	
Descripción: El conductor seleccione la opción iniciar carrera	

Tabla 60. Tarjeta de colaboración HU12 Iniciar sesión

Usuario	
Responsabilidades	**Colaboradores**
• +getPhone() • +getName() • +getPhoto()	• DataContex • BaseActivit • Login

Tabla 61. Historia de Usuario Terminar carrera

Historia de usuario
Número: HU13
Programador Responsable: Team Development
Nombre de la historia: Terminar carrera
Nivel de riesgo de desarrollo: Alta
Peso de la historia: 13 puntos de la historia
Tiempo estimado: 22 Horas
Perspectiva del producto **Como:** Conductor **Se Requiere:** Seleccionar la opción terminar carrera **Para:** Dar por terminada la carrera
Criterios de aceptación • Conductor notifique la finalización de la carrera • Enviar información al Backend

Actividades	N° de horas de implementación
• Interfaz gráfica • Terminar carrera	**7** **15**

Tabla 62. Tarea de Ingeniería Interfaz gráfica

TAREA DE INGENIERÍA	
Número de Tarea: 1	**HU13:** Terminar carrera
Nombre de Tarea: Interfaz gráfica	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 13 puntos de historia
Tiempo de implementación estimado: 7 Horas	
Fecha Inicio: 07/03/2016	**Fecha Fin:** 25/03/2016
Programador Responsable: Team Development	
Descripción: Crear interfaz gráfica para que el conductor seleccione la opción terminar carrera	

Tabla 63. Tarea de Ingeniería Terminar carrera

TAREA DE INGENIERÍA	
Número de Tarea: 2	**HU13:** Terminar carrera
Nombre de Tarea: Terminar carrera	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 13 puntos de historia
Tiempo de implementación estimado: 15 Horas	
Fecha Inicio: 07/03/2016	**Fecha Fin:** 25/03/2016

Programador Responsable: Team Development
Descripción: El conductor seleccione la opción terminar carrera

Tabla 64. Tarjeta de colaboración HU13 Terminar Carrera

Usuario	
Responsabilidades	**Colaboradores**
• +getPhone() • +getName() • +getPhoto()	• DataContex • BaseActivit • Login

Tabla 65. Historia de Usuario Iniciar sesión

Historia de usuario
Número: HU14
Programador Responsable: Team Development
Nombre de la historia: Iniciar sesión
Nivel de riesgo de desarrollo: Alta
Peso de la historia: 13 puntos de la historia
Tiempo estimado: 10 Horas
Perspectiva del producto **Como:** Usuario **Se Requiere:** Iniciar Sesión **Para:** Acceder a las funcionalidades de la aplicación móvil

Criterios de aceptación • Iniciar sesión con correo y contraseña	
Actividades	**N° de horas de implementación**
• Interfaz gráfica • Enviar información al Backend	**5** **5**

Tabla 66. Tarea de Ingeniería interfaz grafica

TAREA DE INGENIERÍA	
Número de Tarea: 1	**HU14:** Iniciar sesión
Nombre de Tarea: Interfaz Grafica	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 13 puntos de historia
Tiempo de implementación estimado: 5 Horas	
Fecha Inicio: 07/03/2016	**Fecha Fin:** 25/03/2016
Programador Responsable: Team Development	
Descripción: Crear interfaz gráfica para iniciar sesión	

Tabla 67. Tarea de ingeniería Enviar información al backend

TAREA DE INGENIERÍA	
Número de Tarea: 2	**HU14:** Iniciar Sesión
Nombre de Tarea: Enviar información al Backend	
Tipo de Tarea: Desarrollo	**Puntos estimados:** 13 puntos de historia
Tiempo de implementación estimado: 5 Horas	

Fecha Inicio: 07/03/2016	**Fecha Fin:** 25/03/2016
Programador Responsable: Team Development	
Descripción: Enviar correo y contraseña de validación al Backend	

Tabla 68. Tarjeta de colaboración HU14 Iniciar sesión

Usuario	
Responsabilidades	**Colaboradores**
• +getPhone() • +getName() • +getPhoto()	• DataContex • BaseActivit • Login

4.4.2 Ejecución del Sprint 3

4.4.2.1 Diseño

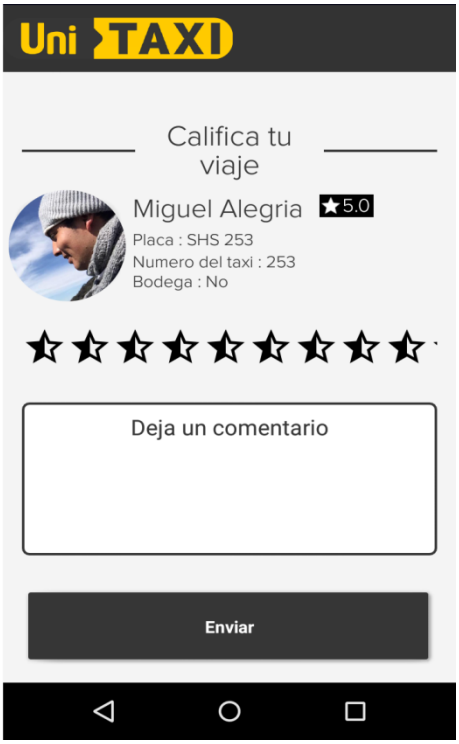


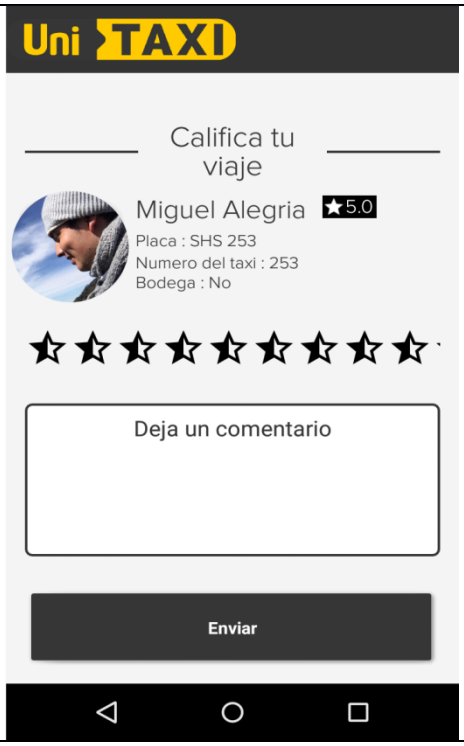
Figura 26. Interfaz gráfica Calificar servicio

Interfaz gráfica de calificación del servicio por parte del usuario al finalizar la carrera o servicio de taxi.

4.4.2.2 Desarrollo

Tabla 69. Evento programado Calificar servicio

Evento Programado Calificar Servicio
Resultados de la calificación



Eventos Programados

```
}

@Override
public void onEvent(Location location) {
    super.onEvent(location);
    myLocation=location;
    if (BdUtils.getApplicationDataContext(this).getUsuariosSet().getMyUser()!=null) {
        ui.showDialog("Loading");
        if (myLocation!= null)
        {
            loginbd(myLocation);
        }

    }
    //ui.dissmisDialgog();



}

private void loginbd(Location myLocation) {
    //ui.showDialog(getResources().getString(R.string.loading));
    cont++;
    if (cont == limit)
    {
        if(BdUtils.getApplicationDataContext(this).getUsuariosSet().getMyUser()!=null) {

            if(myLocation!=null) {
                Gcmi_id = BdUtils.getApplicationDataContext(this).getUsuariosSet().getMyUser().getGmc();
                Map<String, String> param = RestClient.getParamsLogin(BdUtils.getApplicationDataContext(this)
                    .getUsuariosSet().getMyUser().getPass(), myLocation.getLatitude() + "," + myLocation.getLongitude(), Gcmi_id);
                RestClient.get().login(param, this);

            }


        }
    }

}
```

4.4.2.3 Pruebas

Tabla 70. Prueba Funcional Calificar Servicio

Historia de usuario	Calificar Servicio
Propósito	Usuario Califique el servicio de taxi
Requisito	Iniciar Sesión Solicitar Servicio
Pasos	Se ingresa a la aplicación móvil Unitaxi se solicita un servicio de taxi y al terminar la carrera se le solicita al usuario una calificación del servicio.
Resultados esperados	Visualizar la interfaz de calificación del servicio ya que el usuario brindará su comentario y seleccionar número de estrellas, enviada esta información al Backend.

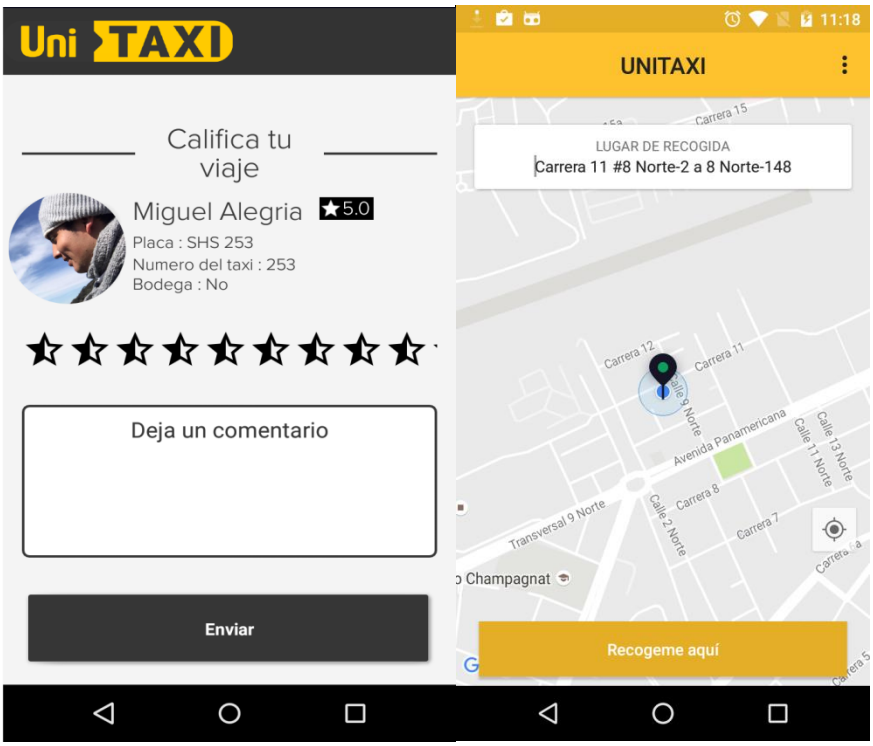


Figura 27. Calificar Servicio y Solicitar servicio [Fuente propia]

4.4.3 Sprint Review 3.

Al finalizar el sprint review el equipo de desarrollo entrega todas las funcionalidades del sprint 3 que son las siguientes:

- Notificar llegada del conductor
- Calificar servicio
- Reportar llegada
- Iniciar carrera
- Terminar carrera
- Iniciar sesión

4.4.4 Sprint Retrospectiva 3.

Tabla 71. Retrospectiva Sprint 3

 • La documentación oficial de Android nos facilitó en el desarrollo del aplicativo. • Uso de *Retrofit* • La experiencia en el desarrollo de aplicaciones móviles. • Usar google cloud messages	 • Tiempo de aprendizaje.
 • Uso de google maps. • Uso google cloud messages	 • Agradecimientos al Scrum master por verificar el cumplimiento de la metodología Scrum.

5. RESULTADOS

La implementación de los aplicativos “Unitaxi” tanto para pasajeros como para conductores, conecta a ciudadanos con la oferta de taxis local y ofrece herramientas para el procesamiento de peticiones en la demanda del servicio de taxi en la ciudad.

Así pues se permite el registro de usuarios a través del aplicativo móvil, el usuario ingresa información personal de acceso o ingresar a través de Facebook; además los conductores podrán iniciar sesión a través de una contraseña asignada que permite gestionar su disponibilidad en el sistema, para poder aceptar las peticiones de los usuarios.

Por otro lado al solicitar un servicio el usuario determina la ubicación donde quiere ser recogido que al momento de ser aceptada por parte del conducto se le enviará al usuario automáticamente la ubicación actual del conductor posteriormente al finalizar el servicio se le envía al usuario la opción de calificación del servicio.

Con los resultados anteriormente mencionados se da cumplimiento a los objetivos específicos:

- Realizar la implementación de las aplicaciones “UniTaxi” (Usuarios) y “UniTaxi manager” (Conductores) para la plataforma Android.
- Diseñar las aplicaciones con el uso de las recomendaciones y especificaciones material design de usabilidad.

Se realizó una prueba el día 03 de diciembre del año 2015 con un grupo de cinco taxistas como se puede ver en el “Anexo B - Información taxistas” y una muestra de nueve usuarios, como se puede ver en el “Anexo C - Pruebas de petición del Taxi”.

Así pues si se compara con el gráfico de la encuesta que se puede ver en el "Anexo D - Servicio de Taxis en Popayán", con los tres minutos de tiempo promedio de muestra.

Figura 28. Resultados de Encuesta "Servicio de Taxis en Popayán"

De ahí que el 42% de las personas encuestadas indican que el tiempo de confirmación del servicio es de cinco a diez minutos, de ahí se concluye que el tiempo de respuesta del servicio se ha reducido en dos minutos si es tomado como tiempo de confirmación según la encuesta el valor de cinco minutos, por esta razón se pude decir que el tiempo que demora en confirmarse el servicio de taxi se ha disminuido en un 40%, por ende se da el cumplimiento al objetivo específico:

Desarrollar un caso de estudio con un grupo de taxistas y usuarios del servicio de taxis en la ciudad de Popayán para evaluar las aplicaciones propuestas.

6. CONCLUSIONES

6.1 CONCLUSIONES

Para el desarrollo del proyecto se realizó una implementación del entorno de trabajo Scrum y la metodología de desarrollo de software Extreme Programming, que permiten el trabajo colaborativo para los miembros del equipo incluyendo desarrolladores, gerentes y clientes. Además de adaptarse a los cambios dados por los clientes en el camino, sin que esto afecte el flujo actual de trabajo, ya que con cada cambio se deben modificar los tiempos y costos, gracias al uso de las buenas practicas recomendadas por estas metodologías. A pesar de que estas metodologías sugieren que los interesados en el proyecto trabajen de la mano, es posible que el mismo equipo de trabajo tome estos roles, respetando la visión de los clientes para lograr la conclusión del proyecto. Estas metodologías otorgan una continua retroalimentación, con esto al equipo se le permite mejorar en cada iteración y responder a los cambios indicados.

El tiempo de aprendizaje para la implementación de los aplicativos móviles para la plataforma Android, con el uso del lenguaje Java y los SDK oficiales proporcionados por Google y terceros, influyo en el tiempo del sprint 1 puesto que las librerías proporcionadas cambian constantemente y su correcta implementación requiere tiempo de aprendizaje que no estaba estimado en el sprint.

Finalmente se puede concluir que la plataforma “Unitaxi”, conecta usuarios con conductores, logra reducir los tiempos de confirmación del servicio solicitado por parte del usuario.

6.2 TRABAJO FUTURO

Como trabajo futuro se visiona cambiar el patrón Modelo Vista Controlador (MVC) utilizado para el desarrollo de los aplicativos móviles por el patrón Model View Presenter (MVP) ya que libera a la vista de lógica del negocio, reduce los llamados al SDK de Android a solo la vista, modulariza la lógica del negocio e implementa una programación funcional.

Por otra parte como modelo de negocio se podrá adicionar una pasarela de pagos que permita la monetización del aplicativo a través de pagos electrónicos.

7. BIBLIOGRAFÍA

[1] "Protesta de taxistas en Popayán por instalación," *Diario el Tiempo*, 2014. .

[2] M. E. Alegría Luligo, Y. F. Jiménez Uribe, and H. A. Satizabal Palacios, "Encuesta Servicio de Taxis en Popayán," Popayan, 2016.

[3] G. F. M, "Popayán necesita más taxis dicen usuarios que se quejan de 'mal servicio," 2014.

[4] "Easy - taxi, car, ridesharing on the App Store." [Online]. Available: https://itunes.apple.com/us/app/easy-taxi-car-ridesharing/id567264775?mt=8.

[5] "Tappsi - Safe Taxis on the App Store." [Online]. Available: https://itunes.apple.com/en/app/tappsi-safe-taxis/id562064313?mt=8.

[6] "Smart Taxi en App Store." [Online]. Available: https://itunes.apple.com/co/app/smart-taxi/id667816497?mt=8.

[7] "CityTaxi en App Store." [Online]. Available: https://itunes.apple.com/co/app/citytaxi/id598987520?mt=8.

[8] B. Ben Elgin, "Google Buys Android for Its Mobile Arsenal," 2005.

[9] "Alliance Overview | Open Handset Alliance." [Online]. Available: http://www.openhandsetalliance.com/oha_faq.html.

[10] "T-Mobile G1 - Full phone specifications." [Online]. Available: http://www.gsmarena.com/t_mobile_g1-2533.php.

[11] N. Gandhewar and R. Sheikh, "Google Android : An Emerging Software Platform For Mobile Devices," *Int. J. Comput. Sci. Eng.*, no. 12, pp. 12–17, 2010.

[12] R. T. Fielding, "Architectural Styles and the Design of Network-based

Software Architectures," *Building*, vol. 54, p. 162, 2000.

[13] "Retrofit: A type-safe REST client for Android and Java," 2015. [Online]. Available: http://square.github.io/retrofit/.

[14] A. Rodriguez, "RESTful Web services: The basics," 2008.

[15] R. Fielding and J. Reschke, "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content."

[16] A. B. Alonso, I. F. Artime, M. Á. Rodríguez, R. G. Baniello, and E. P. S. I. G. I. De Telecomunicación, "Dispositivos móviles," *Telecomunicacion-Universidad De Oviedo*, no. 1, pp. 1–12, 2008.

[17] I. D. Corporation, "Smartphone OS Market Share, 2015 Q2," 2015. [Online]. Available: http://www.idc.com/prodserv/smartphone-os-market-share.jsp.

[18] E. Pérez, "¿Qué es Material Design?," 2014. [Online]. Available: http://www.elandroidelibre.com/2014/11/que-es-material-design.html.

[19] "Agile Unified Process (AUP) - Ingenieria de Soporte Logico." [Online]. Available: https://sites.google.com/site/ingsoportelogico/home/agile-unified-process.

[20] Ambysoft Inc., "The Agile Unified Process," *Ambysoft*, 2005. [Online]. Available: http://www.ambysoft.com/unifiedprocess/agileUP.html.

[21] C. Crystal, "Software Development Methodologies – Lecture 12."

[22] "LEAN-AGILE SOFTWARE DEVELOPMENT Lean-Agile Series LEAN-AGILE SOFTWARE DEVELOPMENT."

[23] "(ootips) Extreme Programming." [Online]. Available: http://ootips.org/xp.html.

[24] J. Joskowicz, "Reglas y prácticas en eXtreme Programming," *Univ. Vigo. España*, pp. 1–22, 2008.

[25] Don Wells, “Extreme Programming: A Gentle Introduction.” [Online]. Available: http://www.extremeprogramming.org/.

[26] L. Da Vinci, “Simplicity Is the Ultimate Sophistication,” *B. Posit. Quotations*, vol. 2, 2006.

[27] “Agile software development through Scrum - Springtimesoft.” [Online]. Available: https://springtimesoft.co.nz/agile-software-development-Scrum/.

[28] “The Scrum Guide™ The Definitive Guide to Scrum: The Rules of the Game,” 2013.

[29] “A Metodologia Scrum para Desenvolvimento de Software.” [Online]. Available: http://metodologiaagil.com/Scrum/.

[30] J. Rasmusson, *The Agile Samurai*. 2010.

[31] “Adobe Illustrator CC | Programa de diseño gráfico.” [Online]. Available: http://www.adobe.com/la/products/illustrator.html.

[32] “Conoce Android Studio | Android Studio.” [Online]. Available: https://developer.android.com/studio/intro/index.html?hl=es-419.

[33] “IntelliJ IDEA the Java IDE.” [Online]. Available: https://www.jetbrains.com/idea/.

[34] “Bitbucket — The Git solution for professional teams.” [Online]. Available: https://bitbucket.org/.

[35] Google Company, “Firebase Cloud Messaging,” *Firebase Cloud Messaging*, 2016. [Online]. Available: https://firebase.google.com/docs/cloud-messaging/.

[36] “Firebase FAQ.” [Online]. Available: https://firebase.google.com/support/faq/#gcm-fcm.

[37] E. Tello-leal, C. M. S. R, and D. A. Tello-leal, “REVISIÓN DE LOS SISTEMAS DE CONTROL DE VERSIONES UTILIZADOS EN EL DESARROLLO DE SOFTWARE,” vol. 3, no. 1, pp. 74–81, 2012.

[38] L. Torvalds and J. Hamano, “Git: Fast version control system,” *URL http//git-scm. com*, 2010.

[39] “Trello.” [Online]. Available: https://Trello.com/.

[40] “What is an aPi?”

[41] D. Chappell, “What Is an Application Platform?,” *Chappell Assoc.*, no. December, 2011.

[42] “Framework Definitions.”

[43] Jules G. McNeff, “The Global Positioning System.”

[44] David J. Eck and Hobart and William Smith Colleges, “Introduction to Programming Using Java.”

[45] E. T. Bray, “The JavaScript Object Notation (JSON) Data Interchange Format.”

[46] L. Torvalds, “Linux: a portable operating system,” *Univ. Helsinki*, p. 52, 1997.

[47] “Cloud Messaging | Google Developers.” [Online]. Available: https://developers.google.com/cloud-messaging/.

[48] R. Fielding, U. C. Irvine, and J. Gettys, “Hypertext Transfer Protocol - HTTP/1.1,” *Ietf*, pp. 1–129, 1999.

[49] E. Flores, “XP - Extreme Programing Ingenieria de Software,” 2016. [Online]. Available: http://ingenieriadesoftware.mex.tl/52753_XP---Extreme-Programing.html.

[50] “What is a Full Stack developer? | Laurence Gellert’s Blog.” [Online].

Available: http://www.laurencegellert.com/2012/08/what-is-a-full-stack-developer/.

[51] M. Rose, “Writing I-Ds and RFCs using XML,” 1999.

[52] P. Studi, I. Komputer, F. Pendidikan, M. Dan, I. Pengetahuan, and U. P. Indonesia, “BAB 1 Pendahuluan.” pp. 1–6, 2010.

ANEXO

Interfaces graficas de la aplicación móvil Usuario

Figura 29. Interfaz de Usuario Registro de Usuario

Interfaz gráfica de inicio de sesión, el usuario podrá acceder a las funcionalidades de la aplicación a través de su correo electrónico y contraseña o dado el caso si desea acceder a través de Facebook, además de la posibilidad de recuperar contraseña y registrarse.

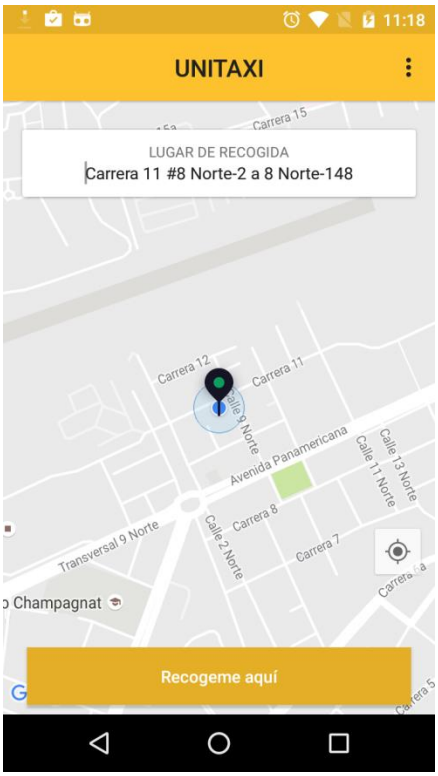


Figura 30. Interfaz de Usuario Solicitar Servicio [Fuente propia]

Interfaz Gráfica para la solicitud del servicio, acorde a la ubicación del GPS generada por el dispositivo móvil o escribir la dirección donde el usuario desea que lo recojan.

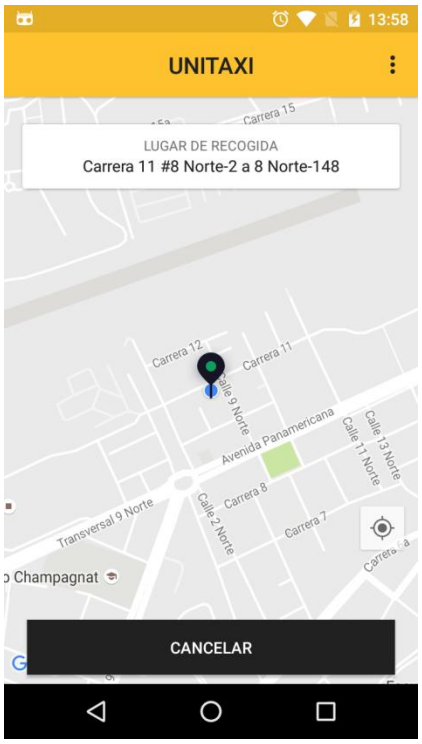


Figura 31. Interfaz de Usuario Cancelar Servicio [Fuente propia]

Interfaz que le permite al usuario cancelar el servicio solicitado.

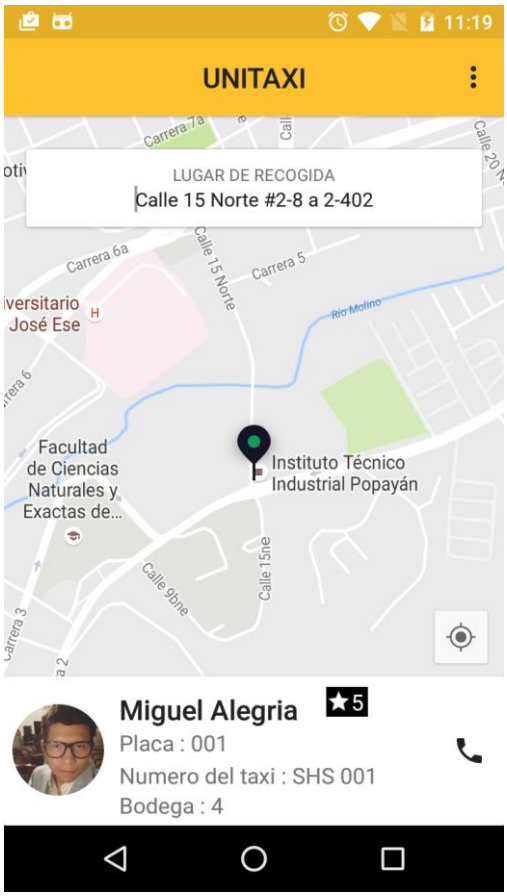


Figura 32. Interfaz Información del Conductor

Interfaz gráfica que proporciona la información del conductor que acepto el servicio, la calificación que tiene y adicionalmente permitirle al usuario realizar una llamada al conductor.

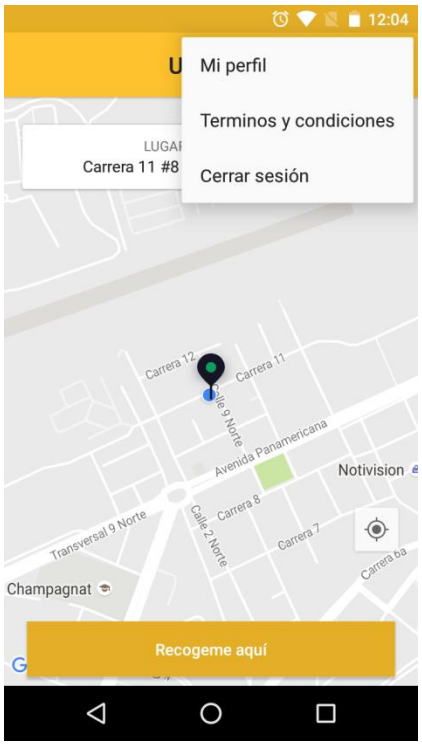


Figura 33. Interfaz gráfica Opciones

Interfaz gráfica permite ver el perfil de usuario, términos y condiciones, además de cerrar sesión.

Figura 34. Interfaz de Usuario Calificar Servicio [Fuente propia]

Interfaz gráfica que permite visualizar la información del conductor y posteriormente dejar un comentario sobre el servicio y una calificación con estrellas.

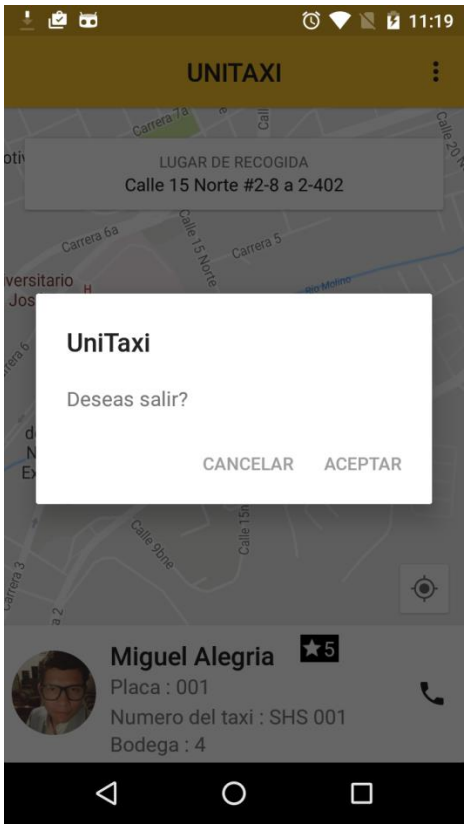


Figura 35. Interfaz gráfica salir de la aplicación.

Interfaz gráfica que le permite al usuario salir de aplicación.

Interfaces graficas de la aplicación móvil Taxista

Figura 36. Interfaz de Taxista Inicio de Sesión [Fuente propia]

Interfaz gráfica para iniciar sesión el conductor, con la contraseña previamente asignada.

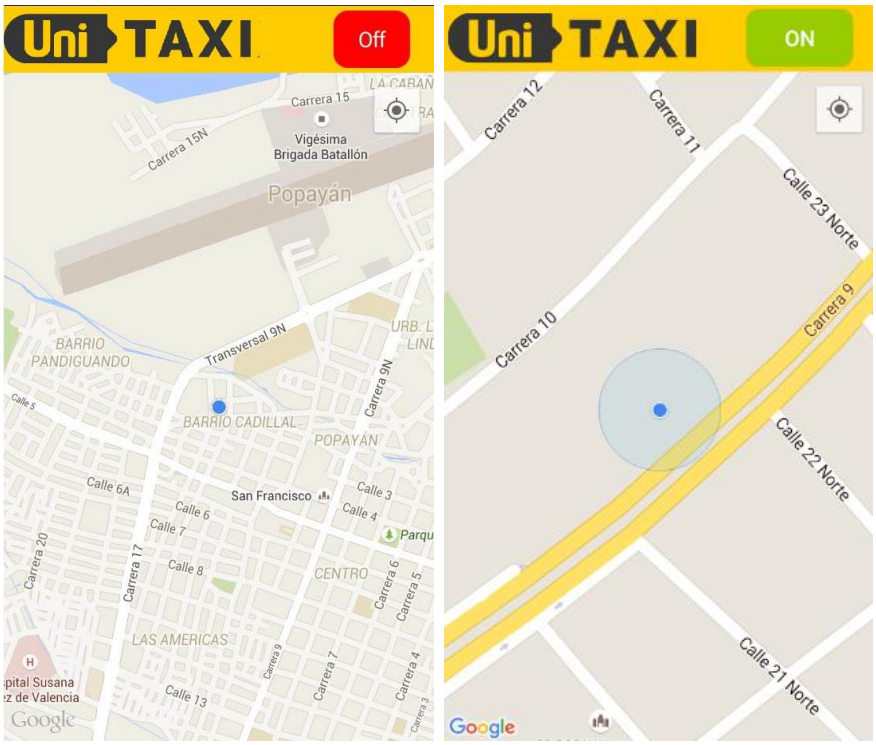


Figura 37. Interfaz Estado del Conductor [Fuente propia]

Interfaces gráficas para visualizar el estado del conductor OFF (Inactivo) Y ON (activo).

Figura 38. Interfaz de Taxista Aceptar Servicio [Fuente propia]

Interfaz gráfica notificación del servicio al conductor para que sea aceptado.

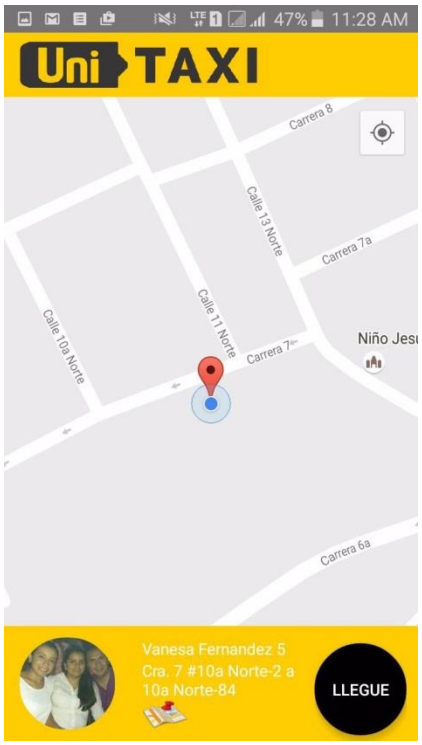


Figura 39. Interfaz de Taxistas Reportar Llegada [Fuente propia]

Interfaz gráfica para que el conductor reporte la llegada al usuario que solicito el servicio de taxi.

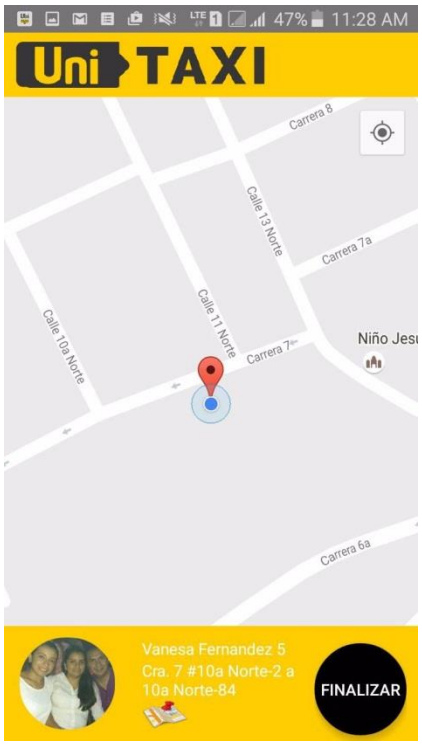


Figura 40. Interfaz gráfica Finalizar Servicio

Interfaz gráfica permite al conductor finalizar el servicio.