UNIVERSIDAD TECNOLÓGICA
DEL CENTRO DE VERACRUZ

Reporte Final de Estadía

Victor Gerardo Texcahua Bautista

Turitaxiac Admin

Av. Universidad No. 350, Carretera Federal Cuitláhuac - La Tinaja
Congregación Dos Caminos, C.P. 94910. Cuitláhuac, Veracruz
Tel. 01 (278) 73 2 20 50
www.utcv.edu.mx

VERACRUZ Gobierno del Estado | SEV Estado de Veracruz | VER Educación Secretaría de Educación | SEP | DET Dirección de Educación Tecnológica del Estado de Veracruz

Universidad Tecnológica Del Centro de Veracruz

Ingeniería en Tecnologías de la Información

Aplicación de Escritorio

Turitaxiac Admin

Alumno:

Víctor Gerardo Texcahua Bautista

Asesor Industrial:

Arturo Merino Castillo

Asesor Académico:

Sergio Velázquez Bonilla

Cuitláhuac Ver a 28 de Abril del 2017.

1 Contenido

Av. Universidad No. 350, Carretera Federal Cuitláhuac - La Tinaja
Congregación Dos Caminos, C.P. 94910. Cuitláhuac, Veracruz
Tel. 01 (278) 73 2 20 50
www.utcv.edu.mx

ÍNDICE DE TABLAS

Av. Universidad No. 350, Carretera Federal Cuitláhuac - La Tinaja
Congregación Dos Caminos, C.P. 94910. Cuitláhuac, Veracruz
Tel. 01 (278) 73 2 20 50
www.utcv.edu.mx

ÍNDICE ILUSTRACIONES

1 Resumen

Este documento describe el trabajo que se realizará en el proceso de estadía de la carrera de Ingeniería de Tecnologías de la información de la Universidad Tecnológica del Centro de Veracruz. El proyecto consiste en el desarrollo de una aplicación de escritorio a la empresa TuritaxiAc Radio Servicio.

Esta empresa se dedica a brindar servicio de taxi al cliente como también registrando la ubicación de los taxis cada hora, y así mismo el manejo de ingreso monetario que genera cada taxi, la empresa no cuenta con alguna aplicación que le ayude a gestionar toda la información que se lleva diariamente. Por lo cual se desarrollará una aplicación de escritorio para facilitarles el manejo de la información, la aplicación de escritorio contará con control de usuarios que podrán acceder a ella.

Con la implementación de este proyecto la gestión de la información de la empresa se agilizará y así lograra brindar mejor servicio a sus clientes y llevar un mejor control de la información de toda la empresa.

2 Introducción

En la actualidad una aplicación de escritorio es un software diseñado para ayudar a facilitar el trabajo de cualquier individuo que haga uso de ella ayudándole a agilizar el manejo de información del lugar donde labore, como también favoreciendo en el ahorro de tiempo y dinero.

Actualmente la tecnología es algo muy importante y se encuentra desde el más pequeño negocio hasta las más grandes y reconocidas empresas.

La empresa TuritaxiAc radio servicios no cuenta con una aplicación que le facilite la gestión de toda la información que maneja dentro de la empresa como el control de quien puede acceder a la información. Se implementará un sistema que facilite el manejo de la información y control de usuarios.

En este documento se describe solo una parte del proyecto propuesto donde se especifica la problemática y solución propuesta, como los beneficios que este proyecto aportará a la empresa, a su vez el documento cuenta con todo el análisis del sistema como la planeación descrita por medio de diagramas y un cronograma de actividades. También cuenta con una descripción de las tecnologías a usarse para hacer posible el desarrollo del sistema.

3 Descripción de la Problemática

En la empresa TuritaxiAc Radio Servicios se lleva el control de los datos de las unidades al servicio, ubicación, si están en servicio, si están en reparación, si está ocupada o disponible para brindar un servicio, también se lleva un control de kilometraje por cada unidad para saber cuándo se tiene que llevar a servicio de: (Afinación, cambio de aceite, remplazo de neumáticos etc..), se llevan formatos donde registran las cuotas que se cobran por pasar viajes a cada unidad diariamente, adeudos y multas por retrasarse en pagos. Llevan el registro de los clientes con todos sus datos (Dirección, referencias e historial de servicios), también manejan una ruleta donde asignan viajes a taxistas por turnos, se lleva un control de servicio de paquetería (Destino, cliente, hora límite para entregar, cuota a cobrar), manejan un registro de publicidad que lleva cada taxi como a la empresa que se le brinda la publicidad para esto manejan los datos (pago por mes, vigencia de publicidad y datos de cada empresa), como el pago de la cuota diaria, Esto provoca que el servicio que se le brinda al cliente cuando solicita el servicio de taxi sea muy tardado, por no poder saber que taxi está disponible ya que es muy difícil consultar

en las hojas, como también a la hora de realizar el pago a los trabajadores por comisión, es difícil buscar los ingresos por cada unidad.

Esto ocasiona que la información no se encuentre disponible y sea difícil de ubicar algún dato que se solicite. Como también no se lleva un control de quien puede consultar la información, esto representa un gran problema en caso de que llegase a extraviarse o sufrir daños las hojas por derrame de líquidos etc., no podría recuperarse la información que es de suma importancia para la empresa en muchos casos la información no es muy legible.

4 Objetivos

4.1 Objetivo General

Desarrollar un Aplicación de escritorio que facilite el manejo de toda la información de la empresa TuritaxiAc, permitiendo llevar el control de los usuarios que ingresan al sistema, control de servicios que brinda la empresa, como el almacenamiento de los datos de las unidades al servicio y sus conductores.

4.2 Objetivos específicos

- Crear los diagramas UML
- Diseñar los mockups de los módulos que contendrá la aplicación.
- Crear las interfaces de la aplicación.
- Crear las hojas de estilo (css) para ajustar los colores solicitados por el cliente.
- Crear diccionario de datos de la base de datos.
- Realizar el diseño de la base de datos.

5 Justificación

Hoy en día una aplicación de escritorio es suma importancia para el manejo de información y para estar a la vanguardia, el proyecto se justifica por la falta de un software que lleve un control preciso de toda la información de la empresa TuritaxiAC ya que en la actualidad la empresa maneja toda la información en hojas, esto tiene un impacto negativo ya que provoca pérdida de tiempo cuando se quiere consultar algún dato que se necesite y no se tiene a la mano.

Se implementará una aplicación que permita la gestión de la información y control de usuarios, permitiendo una fácil consulta, agilizando los procesos que se llevan diariamente así ahorrando tiempo, esfuerzo y dinero, brindando un mejor servicio al

cliente teniendo la información disponible para consultarla a cualquier hora y sin necesidad de buscarla hoja por hoja ya que la aplicación facilitará la búsqueda.

El desarrollo de este proyecto tendrá un gran impacto positivo en el área de administración de la empresa y servicios.

6 Alcance

La aplicación de escritorio está planeada a desarrollarse en 8 meses, 4 meses (Septiembre- Diciembre) se realizará un el protocolo de investigación del proyecto. En los últimos 4 meses (Enero- Abril) en el proceso de estadía se llevara a cabo el desarrollo de la aplicación como su implementación.

La aplicación contará con un módulo de login para el control de los usuarios permitidos, un módulo donde se podrá dar de alta las unidades y sus características, un módulo donde se dará de alta los conductores y se les asignará una unidad, un módulo donde se podrá llevar el registro de los movimientos de las unidades y su estado activo o inactivo. Un módulo donde se registrarán los clientes y los servicios que se le brindan, un módulo donde se lleve el control del pago diario que realizan los taxistas y así mismo el historial de pagos como multas por retrasarse, un módulo donde se registrarán los servicios de paquetería donde gestionarán la ora límite de entrega y el destino, un módulo donde de forma de ruleta se puedan ir asignando servicios a cada unidad, se contará con un módulo donde se lleve el control de la publicidad y el tiempo de vigencia como la información de la empresa que solicita el servicio de publicidad, se llevara el control de el kilometraje por cada unidad que cuando sumen cierto kilometraje nos mande una alerta. Se podrá realizar respaldos de la base de datos diariamente, por último se podrá imprimir reportes.

La aplicación podrá ejecutarse solo en computadoras que cuenten con Windows vista y versiones posteriores, para el buen funcionamiento de la aplicación la computadora deberá tener estas características mínimas: Disco duro de 80 gb, 2gb de memoria RAM, procesador a 2.7 GHz.

No contara con localización de las unidades vía GPS

7 Marco Teórico

7.1 Metodología XP

METODOLOGÍA XP Según Kent Beck 1999 ORIGEN DE LA METODOLOGÍA XP La programación extrema o Xtreme Programming (XP) es una metodología de desarrollo de la ingeniería de software formulada por Kent Beck, autor del primer libro sobre la materia, Extreme Programming Explained: Embrace Change (1999). Es el más destacado de los procesos ágiles de desarrollo de software. Al igual que éstos, la programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad. Los defensores de XP consideran que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos. Se puede considerar la programación extrema como la adopción de las mejores metodologías de desarrollo de acuerdo a lo que se pretende llevar a cabo con el proyecto, y aplicarlo de manera dinámica durante el ciclo de vida del software.

CARACTERÍSTICAS DE LA METODOLOGÍA XP

- Se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad.
- Se aplica de manera dinámica durante el ciclo de vida del software.
- Es capaz de adaptarse a los cambios de requisitos. Los individuos e interacciones son más importantes que los procesos y herramientas.
- Al individuo y las interacciones del equipo de desarrollo sobre el proceso y las herramientas.

La gente es el principal factor de éxito de un proyecto software. Es más importante construir un buen equipo que construir el entorno. Muchas veces se comete el error de construir primero el entorno y esperar que el equipo se adapte automáticamente. Es mejor crear el equipo y que éste configure su propio entorno de desarrollo en base a sus necesidades.

- software que funciona más que conseguir una buena documentación.

La regla a seguir es no producir documentos a menos que sean necesarios de forma inmediata para tomar una decisión importante. Estos documentos deben ser cortos y centrarse en lo fundamental.

- La colaboración con el cliente es más importante que la negociación de contratos.
- La colaboración con el cliente más que la negociación de un contrato.

Pasos De La Metodología Xp

Los Pasos fundamentales inmersos en las fases del método son:

Desarrollo iterativo e incremental: Pequeñas mejoras, unas tras otras. Pruebas unitarias continuas: Son frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión. Se aconseja escribir el código de la prueba antes de la codificación.

Programación en parejas: Se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto. Se supone que la mayor calidad del código escrito de esta manera -el código es revisado y discutido mientras se escribe- es más importante que la posible pérdida de productividad inmediata.

Frecuente integración del equipo de programación con el cliente o usuario: Se recomienda que un representante del cliente trabaje junto al equipo de desarrollo.

Corrección de todos los errores antes de añadir nueva funcionalidad. Hacer entregas frecuentes.

Refactorización del código: Es decir, reescribir ciertas partes del código para aumentar su legibilidad y Mantenibilidad pero sin modificar su comportamiento. Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo.

Propiedad del código compartido: en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve el que todo el personal pueda corregir y extender cualquier parte del proyecto. Las frecuentes pruebas de regresión garantizan que los posibles errores serán detectados.

Simplicidad del código: es la mejor manera de que las cosas funcionen. Cuando todo funcione se podrá añadir funcionalidad si es necesario. La programación extrema apuesta que es más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo.

La simplicidad y la comunicación son extraordinariamente complementarias. Con más comunicación resulta más fácil identificar qué se debe y qué no se debe hacer. Cuanto

más simple es el sistema, menos tendrá que comunicar sobre éste, lo que lleva a una comunicación más completa, especialmente si se puede reducir el equipo de programadores. (Bustamante Dayana, 2014)

7.2 MySQL

MySQL es el servidor de bases de datos relacionales más popular, desarrollado y proporcionado por MySQL AB. MySQL AB es una empresa cuyo negocio consiste en proporcionar servicios en torno al servidor de bases de datos MySQL.

MySQL es un sistema de administración de bases de datos
Una base de datos es una colección estructurada de datos. La información que puede almacenar una base de datos puede ser tan simple como la de una agenda, un contador, o un libro de visitas, o tan vasta como la de una tienda en línea, un sistema de noticias, un portal, o la información generada en una red corporativa. Para agregar, accesar, y procesar los datos almacenados en una base de datos, se necesita un sistema de administración de bases de datos, tal como MySQL.

MySQL es un sistema de administración de bases de datos relacionales
Una base de datos relacional almacena los datos en tablas separadas en lugar de poner todos los datos en un solo lugar. Esto agrega velocidad y flexibilidad. Las tablas son enlazadas al definir relaciones que hacen posible combinar datos de varias tablas cuando se necesitan consultar datos. La parte SQL de "MySQL" significa "Lenguaje Estructurado de Consulta", y es el lenguaje más usado y estandarizado para accesar a bases de datos relacionales.

MySQL es Open Source

Open Source significa que la persona que quiera puede usar y modificar MySQL. Cualquiera puede descargar el software de MySQL de Internet y usarlo sin pagar por ello. Inclusive, cualquiera que lo necesite puede estudiar el código fuente y cambiarlo de acuerdo a sus necesidades. MySQL usa la licencia GPL (Licencia Pública General GNU), para definir qué es lo que se puede y no se puede hacer con el software para diferentes situaciones. Sin embargo, si uno está incómodo con la licencia GPL o tiene la necesidad de incorporar código de MySQL en una aplicación comercial es posible comprar una versión de MySQL con una licencia comercial. Para mayor información, ver la página oficial de MySQL en la cual se proporciona mayor información acerca de los tipos de licencias.

¿Por qué usar MySQL?

El servidor de bases de datos MySQL es muy rápido, seguro, y fácil de usar. Si eso es lo que se está buscando, se le debe dar una oportunidad a MySQL. Se pueden encontrar comparaciones de desempeño con algunos otros manejadores de bases de datos en la página de MySQL.

El servidor MySQL fue desarrollado originalmente para manejar grandes bases de datos mucho más rápido que las soluciones existentes y ha estado siendo usado exitosamente en ambientes de producción sumamente exigentes por varios años. Aunque se encuentra en desarrollo constante, el servidor MySQL ofrece hoy un conjunto rico y útil de funciones. Su conectividad, velocidad, y seguridad hacen de MySQL un servidor bastante apropiado para accesar a bases de datos en Internet.

Algunos detalles técnicos de MySQL

El software de bases de datos MySQL consiste de un sistema cliente/servidor que se compone de un servidor SQL multihilo, varios programas clientes y bibliotecas, herramientas administrativas, y una gran variedad de interfaces de programación (APIs). Se puede obtener también como una biblioteca multihilo que se puede enlazar dentro de otras aplicaciones para obtener un producto más pequeño, más rápido, y más fácil de manejar. Para obtener información técnica más detallada, es necesario consultar la guía de referencia de MySQL. (Bravo, 2006).

7.3 Java

Sun Microsystems desarrolló, en 1991, el lenguaje de programación orientado a objetos que se conoce como Java. El objetivo era utilizarlo en un set-top box, un tipo de dispositivo que se encarga de la recepción y la descodificación de la señal televisiva. El primer nombre del lenguaje fue Oak, luego se conoció como Green y finamente adoptó la denominación de Java.

La intención de Sun era crear un lenguaje con una estructura y una sintaxis similar a C y C++, aunque con un modelo de objetos más simple y eliminando las herramientas de bajo nivel.

Los pilares en los que se sustenta Java son cinco: la programación orientada a objetos, la posibilidad de ejecutar un mismo programa en diversos sistemas operativos, la inclusión por defecto de soporte para trabajo en red, la opción de ejecutar el código en sistemas remotos de manera segura y la facilidad de uso.

Lo habitual es que las aplicaciones Java se encuentren compiladas en un bytecode (un fichero binario que tiene un programa ejecutable), aunque también pueden estar compiladas en código máquina nativo.

Sun controla las especificaciones y el desarrollo del lenguaje, los compiladores, las máquinas virtuales y las bibliotecas de clases a través del Java Community Process. En los últimos años, la empresa (que fue adquirida por Oracle) ha liberado gran parte de las tecnologías Java bajo la licencia GNU GPL.

La aplicación de Java es muy amplia. El lenguaje se utiliza en una gran variedad de dispositivos móviles, como teléfonos y pequeños electrodomésticos. Dentro del ámbito de Internet, Java permite desarrollar pequeñas aplicaciones (conocidas con el nombre de applets) que se incrustan en el código HTML de una página, para su directa ejecución desde un navegador; cabe mencionar que es necesario contar con el plug-in adecuado para su funcionamiento, pero la instalación es liviana y sencilla. (Julián Pérez Porto, 2010).

7.4 HeidiSQL

MySQL es una de las bases de datos relacionales más utilizadas en Internet. Potente, rápida y con poco gasto de recursos, esta herramienta, que se distribuye bajo licencia GLP, es un imprescindible para muchos diseñadores web. Aunque es recomendable aprender a trabajar SQL desde la línea de comandos, los no expertos tienen alternativas para trabajar con sus bases de datos a golpe de clic.

HeidiSQL es una interfaz gráfica opensource para la gestión simplificada de tus bases de datos MySQL locales y/o remotas. Esta ligera aplicación gratuita ofrece una gran cantidad de funcionalidades para que la gestión de tus bases de datos se convierta en una tarea asequible.

Entre otras muchas funciones, HeidiSQL te permite:

– La gestión y consulta de tus bases y tablas de datos a partir de una interfaz Windows ergonómica.
– La importación de datos desde archivos de texto.
– La exportación de tablas de datos CSV, HTML y XML.
– La sincronización de tablas entre dos bases de datos.
– La gestión de los distintos usuarios y sus privilegios.
– La edición sencilla de tablas de datos.

En resumen, una interfaz completa, intuitiva, fácil de utilizar y realmente útil para los desarrolladores Web. (Maria, 2007)

7.5 JavaFX

JavaFX es una plataforma de software que permite crear y ejecutar aplicaciones web que pueden aplicarse en una gran variedad de dispositivos.

JavaFX pertenece a las tecnologías de Sun Microsystems, adquirida por Oracle Corporation, sirve para la creación de *Rich Internet Applications* (RIAs), esto es, aplicaciones web que tienen las características y capacidades de aplicaciones de escritorio, incluyendo aplicaciones multimedia interactivas. Las tecnologías incluidas bajo la denominación JavaFX son JavaFX Script y JavaFX Mobile, aunque hay más productos JavaFX planeados.

Historia:
En la conferencia JavaOne celebrada en San Francisco en 2007, Sun Microsystems presentó la plataforma JavaFX para ayudar a los desarrolladores de contenido y desarrolladores de aplicaciones a crear aplicaciones ricas en contenido para dispositivos móviles, escritorios, televisores y otros dispositivos de consumo. La oferta inicial consistió en la plataforma JavaFX Mobile y el lenguaje JavaFX Script.

El 4 de diciembre del 2008 Sun Microsystems lanzó JavaFx 1.0, así como el plugin de JavaFx para NetBeans.

El 12 de febrero del 2009 se lanzó la versión 1.1 de JavaFx.

La versión 1.2 de JavaFx lanzada en junio del 2009 permitía la creación de aplicaciones de escritorio, navegador, teléfono móvil, televisión, consolas de juegos y Blue-ray, la

versión 2.0 es, hasta ahora la versión más reciente y puede ser descargada desde su sitio oficial.

Características:

La creación de aplicaciones JavaFX se hace a través del lenguaje JavaFX Script.

Las aplicaciones de escritorio pueden ejecutarse en Windows XP, Windows Vista, Windows 7, Mac OS, GNU/Linux y OpenSolaris.

Al estar integrado con el JRE (Java Runtime Environment), las aplicaciones de JavaFX se pueden ejecutar en cualquier navegador que posean el JRE, como así también móviles que dispongan de Java ME.

El compilador de JavaFx y el plugin para NetBeans se encuentran bajo licencia GPL

Ventajas:

- Funciona en cualquier navegador donde se encentre instalado JRE (*Java Runtime Enviroment*).
- Software multiplataforma.
- Java se encuentra bajo licencia GPL.
- Dado que la plataforma JavaFX está escrito en Java, los desarrolladores de Java pueden aprovechar sus habilidades existentes y las herramientas para desarrollar aplicaciones JavaFX, como librerías, métodos o variables.
- Para desarrollar aplicaciones JavaFx se puede utilizar cualquier editor de textos o cualquier entorno de desarrollo integrado (IDE) que soporte lenguaje Java, como NetBeans, Eclipse, Oracle JDeveloper, o Intellij IDEA. (Alexander, 2011)

Desventajas:

Al ser un lenguaje de Script agrega un factor de ralentización muy importante, por eso el desempeño no es muy óptimo.

8 Metodología De Desarrollo

Basarse en una metodología para el buen desarrollo de un proyecto es de suma importancia, para este proyecto se eligió una metodología ágil XP Extreme Programing ya que el desarrollo del proyecto será en 4 meses, esta metodología a comparación de otras fue la que más se adaptó al tiempo del proyecto ya que solo se generan pocos artefactos

como está dirigida a grupos pequeños de trabajo, y principalmente en comunicación constante con el cliente.

8.1 Comparación de metodologías agiles y tradicionales

Metodología Ágil	Metodología Tradicional
Pocos Artefactos. El modelado es prescindible, modelos desechables.	Más Artefactos. El modelado es esencial, mantenimiento de modelos
Pocos Roles, más genéricos y flexibles	Más Roles, más específicos
No existe un contrato tradicional, debe ser bastante flexible	Existe un contrato prefijado
Cliente es parte del equipo de desarrollo (además in-situ)	El cliente interactúa con el equipo de desarrollo mediante reuniones
Orientada a proyectos pequeños. Corta duración (o entregas frecuentes), equipos pequeños (< 10 integrantes) y trabajando en el mismo sitio	Aplicables a proyectos de cualquier tamaño, pero suelen ser especialmente efectivas/usadas en proyectos grandes y con equipos posiblemente dispersos
La arquitectura se va definiendo y mejorando a lo largo del proyecto	Se promueve que la arquitectura se defina tempranamente en el proyecto
Énfasis en los aspectos humanos: el individuo y el trabajo en equipo	Énfasis en la definición del proceso: roles, actividades y artefactos
Basadas en heurísticas provenientes de prácticas de producción de código	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo
Se esperan cambios durante el proyecto	Se espera que no ocurran cambios de gran impacto durante el proyecto

Tabla 1 Comparación de metodologías agiles

8.2 Revisión de metodologías

Aunque los creadores e impulsores de las metodologías ágiles más populares han suscrito el manifiesto ágil y coinciden con los principios enunciados anteriormente, cada metodología tiene características propias y hace hincapié en algunos aspectos más

específicos. A continuación se resumen dichas metodologías ágiles, dejando el análisis más detallado de XP para la siguiente sección.

- **SCRUM** Desarrollada por Ken Schwaber, Jeff Sutherland y Mike Beedle. Define un marco para la gestión de proyectos, que se ha utilizado con éxito durante los últimos 10 años. Está especialmente indicada para proyectos con un rápido cambio de requisitos. Sus principales características se pueden resumir en dos. El desarrollo de software se realiza mediante iteraciones, denominadas sprints, con una duración de 30 días. El resultado de cada sprintes un incremento ejecutable que se muestra al cliente. La segunda característica importante son las reuniones a lo largo proyecto. Éstas son las verdaderas protagonistas, especialmente la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración.

- **Crystal Methodologies** Se trata de un conjunto de metodologías para el desarrollo de software caracterizadas por estar centradas en las personas que componen el equipo (de ellas depende el éxito del proyecto) y la reducción al máximo del número de artefactos producidos. Han sido desarrolladas por Alistair Cockburn. El desarrollo de software se considera un juego cooperativo de invención y comunicación, limitado por los recursos a utilizar. El equipo de desarrollo es un factor clave, por lo que se deben invertir esfuerzos en mejorar sus habilidades y destrezas, así como tener políticas de trabajo en equipo definidas. Estas políticas dependerán del tamaño del equipo, estableciéndose una clasificación por colores, por ejemplo Crystal Clear (3 a 8 miembros) y Crystal Orange (25 a 50 miembros).

- **Dynamic Systems Development Method** (**DSDM**) Define el marco para desarrollar un proceso de producción de software. Nace en 1994 con el objetivo el objetivo de crear una metodología RAD unificada. Sus principales características son: es un proceso iterativo e incremental y el equipo de desarrollo y el usuario trabajan juntos. Propone cinco fases: estudio viabilidad, estudio del negocio, modelado funcional, diseño y construcción, y finalmente implementación. Las tres últimas son iterativas, además de existir realimentación a todas las fases.

- **Adaptive Software Development** (**ASD**) Su impulsor es Jim Highsmith. Sus principales características son: iterativo, orientado a los componentes software más que a las tareas y tolerante a los cambios. El ciclo de vida que propone tiene tres fases esenciales: especulación, colaboración y aprendizaje. En la primera de

ellas se inicia el proyecto y se planifican las características del software; en la segunda desarrollan las características y finalmente en la tercera se revisa su calidad, y se entrega al cliente. La revisión de los componentes sirve para aprender de los errores y volver a iniciar el ciclo de desarrollo.

- **Feature-Driven Development** (**FDD**) Define un proceso iterativo que consta de 5 pasos. Las iteraciones son cortas (hasta 2 semanas). Se centra en las fases de diseño e implementación del sistema partiendo de una lista de características que debe reunir el software. Sus impulsores son Jeff De Luca y Peter Coad.

- **Lean Development** (**LD**) [Definida por Bob Charette's a partir de su experiencia en proyectos con la industria japonesa del automóvil en los años 80 y utilizada en numerosos proyectos de telecomunicaciones en Europa. En LD, los cambios se consideran riesgos, pero si se manejan adecuadamente se pueden convertir en oportunidades que mejoren la productividad del cliente. Su principal característica es introducir un mecanismo para implementar dichos cambios. (Patricio Letelier, 2006)

8.3 Metodología XP Extreme Programing

1ª Fase: Planificación del proyecto.

Historias de usuario.

- Release planning.
- Iteraciones:
- Velocidad del proyecto.
- Programación en pareja.
- Reuniones diarias.

• 2ª Fase: Diseño.

Diseños simples.

Glosarios de términos.

Riesgos.

Funcionalidad extra.

Tarjetas C.R.C.

• 3ª Fase: Codificación.

• 4ª Fase: Pruebas.

El uso de los test en X.P es el siguiente.

Test de aceptación.

8.3.1 1ª Fase: Planificación del proyecto.

Historias de usuario: El primer paso de cualquier proyecto que siga la metodología X.P es definir las historias de usuario con el cliente. Las historias de usuario tienen la misma finalidad que los casos de uso pero con algunas diferencias: Constan de 3 o 4 líneas escritas por el cliente en un lenguaje no técnico sin hacer mucho hincapié en los detalles; no se debe hablar ni de posibles algoritmos para su implementación ni de diseños de base de datos adecuados, etc. Son usadas para estimar tiempos de desarrollo de la parte de la aplicación que describen. También se utilizan en la fase de pruebas, para verificar si el programa cumple con lo que especifica la historia de usuario. Cuando llega la hora de implementar una historia de usuario, el cliente y los desarrolladores se reúnen para concretar y detallar lo que tiene que hacer dicha historia. El tiempo de desarrollo ideal para una historia de usuario es entre 1 y 3 semanas.

Release planning: .Después de tener ya definidas las historias de usuario es necesario crear un plan de publicaciones, en inglés "Release plan", donde se indiquen las historias de usuario que se crearán para cada versión del programa y las fechas en las que se publicarán estas versiones. Un "Release plan" es una planificación donde los desarrolladores y clientes establecen los tiempos de implementación ideales de las historias de usuario, la prioridad con la que serán implementadas y las historias que serán implementadas en cada versión del programa. Después de un "Release plan" tienen que estar claros estos cuatro factores: los objetivos que se deben cumplir (que son principalmente las historias que se deben desarrollar en cada versión), el tiempo que tardarán en desarrollarse y publicarse las versiones del programa, el número de personas que trabajarán en el desarrollo y cómo se evaluará la calidad del trabajo realizado. (*Release plan: Planificación de publicaciones).

Iteraciones: Todo proyecto que siga la metodología X.P. se ha de dividir en iteraciones de aproximadamente 3 semanas de duración. Al comienzo de cada iteración los clientes deben seleccionar las historias de usuario definidas en el "Release planning" que serán implementadas. También se seleccionan las historias de usuario que no pasaron el test de aceptación que se realizó al terminar la iteración anterior. Estas historias de usuario

son divididas en tareas de entre 1 y 3 días de duración que se asignarán a los programadores.

Velocidad del proyecto: La velocidad del proyecto es una medida que representa la rapidez con la que se desarrolla el proyecto; estimarla es muy sencillo, basta con contar el número de historias de usuario que se pueden implementar en una iteración; de esta forma, se sabrá el cupo de historias que se pueden desarrollar en las distintas iteraciones. Usando la velocidad del proyecto controlaremos que todas las tareas se puedan desarrollar en el tiempo del que dispone la iteración. Es conveniente reevaluar esta medida cada 3 o 4 iteraciones y si se aprecia que no es adecuada hay que negociar con el cliente un nuevo "Release Plan".

Programación en pareja: La metodología X.P. aconseja la programación en parejas pues incrementa la productividad y la calidad del software desarrollado. El trabajo en pareja involucra a dos programadores trabajando en el mismo equipo; mientras uno codifica haciendo hincapié en la calidad de la función o método que está implementando, el otro analiza si ese método o función es adecuado y está bien diseñado. De esta forma se consigue un código y diseño con gran calidad.

Reuniones diarias: Es necesario que los desarrolladores se reúnan diariamente y expongan sus problemas, soluciones e ideas de forma conjunta. Las reuniones tienen que ser fluidas y todo el mundo tiene que tener voz y voto.

2ª Fase: Diseño.

Diseños simples: La metodología X.P sugiere que hay que conseguir diseños simples y sencillos. Hay que procurar hacerlo todo lo menos complicado posible para conseguir un diseño fácilmente entendible e impleméntable que a la larga costará menos tiempo y esfuerzo desarrollar.

Glosarios de términos: Usar glosarios de términos y una correcta especificación de los nombres de métodos y clases ayudará a comprender el diseño y facilitará sus posteriores ampliaciones y la reutilización del código.

Riesgos: Si surgen problemas potenciales durante el diseño, X.P sugiere utilizar una pareja de desarrolladores para que investiguen y reduzcan al máximo el riesgo que supone ese problema.

Funcionalidad extra: Nunca se debe añadir funcionalidad extra al programa aunque se piense que en un futuro será utilizada. Sólo el 10% de la misma es utilizada, lo que implica que el desarrollo de funcionalidad extra es un desperdicio de tiempo y recursos.

Refactorizar: Refactorizar es mejorar y modificar la estructura y codificación de códigos ya creados sin alterar su funcionalidad. Refactorizar supone revisar de nuevo estos códigos para procurar optimizar su funcionamiento. Es muy común rehusar códigos ya creados que contienen funcionalidades que no serán usadas y diseños obsoletos. Esto es un error porque puede generar código completamente inestable y muy mal diseñado; por este motivo, es necesario refactorizar cuando se va a utilizar código ya creado.

Tarjetas C.R.C.: El uso de las tarjetas C.R.C (Class, Responsabilities and Collaboration) permiten al programador centrarse y apreciar el desarrollo orientado a objetos olvidándose de los malos hábitos de la programación procedural clásica.

Las tarjetas C.R.C representan objetos; la clase a la que pertenece el objeto se puede escribir en la parte de arriba de la tarjeta, en una columna a la izquierda se pueden escribir las responsabilidades u objetivos que debe cumplir el objeto y a la derecha, las clases que colaboran con cada responsabilidad.

8.3.2 3ª Fase: Codificación.

Como ya se dijo en la introducción, el cliente es una parte más del equipo de desarrollo; su presencia es indispensable en las distintas fases de X.P. A la hora de codificar una historia de usuario su presencia es aún más necesaria. No olvidemos que los clientes son los que crean las historias de usuario y negocian los tiempos en los que serán implementadas. Antes del desarrollo de cada historia de usuario el cliente debe especificar detalladamente lo que ésta hará y también tendrá que estar presente cuando se realicen los test que verifiquen que la historia implementada cumple la funcionalidad especificada.

La codificación debe hacerse ateniendo a estándares de codificación ya creados. Programar bajo estándares mantiene el código consistente y facilita su comprensión y escalabilidad.

Crear test que prueben el funcionamiento de los distintos códigos implementados nos ayudará a desarrollar dicho código. Crear estos test antes nos ayuda a saber qué es exactamente lo que tiene que hacer el código a implementar y sabremos que una vez implementado pasará dichos test sin problemas ya que dicho código ha sido diseñado para ese fin. Se puede dividir la funcionalidad que debe cumplir una tarea a programar en pequeñas unidades, de esta forma se crearán primero los test para cada unidad y a continuación se desarrollará dicha unidad, así poco a poco conseguiremos un desarrollo que cumpla todos los requisitos especificados.

Como ya se comentó anteriormente, X.P opta por la programación en pareja ya que permite un código más eficiente y con una gran calidad.

X.P sugiere un modelo de trabajo usando repositorios de código dónde las parejas de programadores publican cada pocas horas sus códigos implementados y corregidos junto a los test que deben pasar. De esta forma el resto de programadores que necesiten códigos ajenos trabajarán siempre con las últimas versiones. Para mantener un código consistente, publicar un código en un repositorio es una acción exclusiva para cada pareja de programadores.

X.P también propone un modelo de desarrollo colectivo en el que todos los programadores están implicados en todas las tareas; cualquiera puede modificar o ampliar una clase o método de otro programador si es necesario y subirla al repositorio de código. El permitir al resto de los programadores modificar códigos que no son suyos no supone ningún riesgo ya que para que un código pueda ser publicado en el repositorio tiene que pasar los test de funcionamiento definidos para el mismo.

La optimización del código siempre se debe dejar para el final. Hay que hacer que funcione y que sea correcto, más tarde se puede optimizar.

X.P afirma que la mayoría de los proyectos que necesiten más tiempo extra que el planificado para ser finalizados no podrán ser terminados a tiempo se haga lo que se haga, aunque se añadan más desarrolladores y se incrementen los recursos. La solución que plantea X.P es realizar un nuevo "Release plan" para concretar los nuevos tiempos de publicación y de velocidad del proyecto.

A la hora de codificar no seguimos la regla de X.P que aconseja crear test de funcionamiento con entornos de desarrollo antes de programar. Nuestros test los obtendremos de la especificación de requisitos ya que en ella se especifican las pruebas que deben pasar las distintas funcionalidades del programa, procurando codificar pensando en las pruebas que debe pasar cada funcionalidad.

8.3.3 4ª Fase: Pruebas.

Uno de los pilares de la metodología X.P es el uso de test para comprobar el funcionamiento de los códigos que vayamos implementando.

El uso de los test en X.P es el siguiente:

Se deben crear las aplicaciones que realizarán los test con un entorno de desarrollo específico para test.

Hay que someter a tests las distintas clases del sistema omitiendo los métodos más triviales.

Se deben crear los test que pasarán los códigos antes de implementarlos; en el apartado anterior se explicó la importancia de crear antes los test que el código.

Un punto importante es crear test que no tengan ninguna dependencia del código que en un futuro evaluará. Hay que crear los test abstrayéndose del futuro código, de esta forma aseguraremos la independencia del test respecto al código que evalúa.

Como se comentó anteriormente los distintos test se deben subir al repositorio de código acompañados del código que verifican. Ningún código puede ser publicado en el repositorio sin que haya pasado su test de funcionamiento, de esta forma, aseguramos el uso colectivo del código (explicado en el apartado anterior).

El uso de los test es adecuado para observar la refactorización. Los test permiten verificar que un cambio en la estructura de un código no tiene por qué cambiar su funcionamiento.

Test de aceptación. Los test mencionados anteriormente sirven para evaluar las distintas tareas en las que ha sido dividida una historia de usuario. Para asegurar el funcionamiento final de una determinada historia de usuario se deben crear "Test de aceptación"; estos test son creados y usados por los clientes para comprobar que las distintas historias de usuario cumplen su cometido.

Al ser las distintas funcionalidades de nuestra aplicación no demasiado extensas, no se harán test que analicen partes de las mismas, sino que las pruebas se realizarán para las funcionalidades generales que debe cumplir el programa especificado en la descripción de requisitos.

9 Modelado UML

9.1 Modelo Entidad-Relación

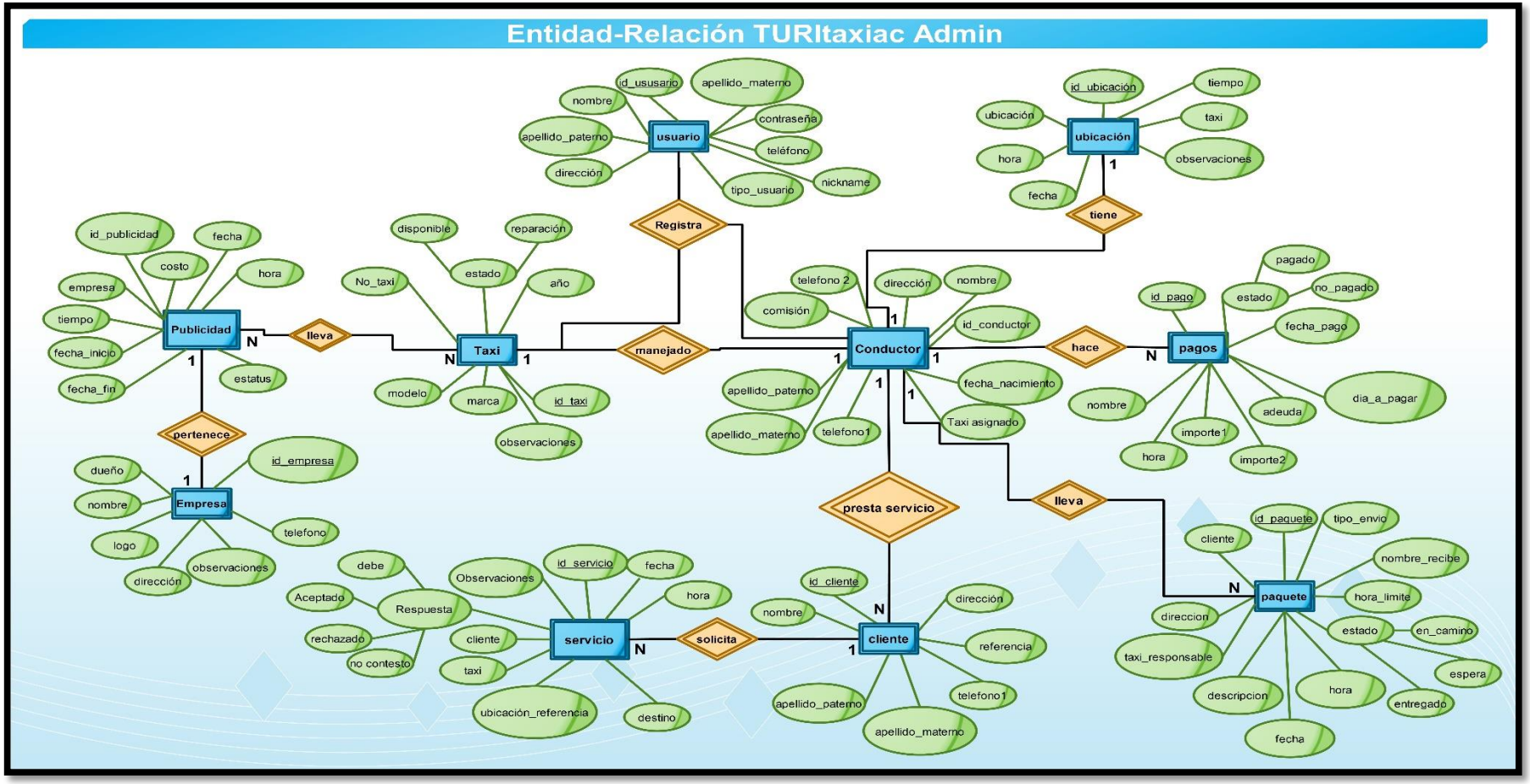

Ilustración 1 Modelo Entidad-relación

9.2 Modelo Relacional

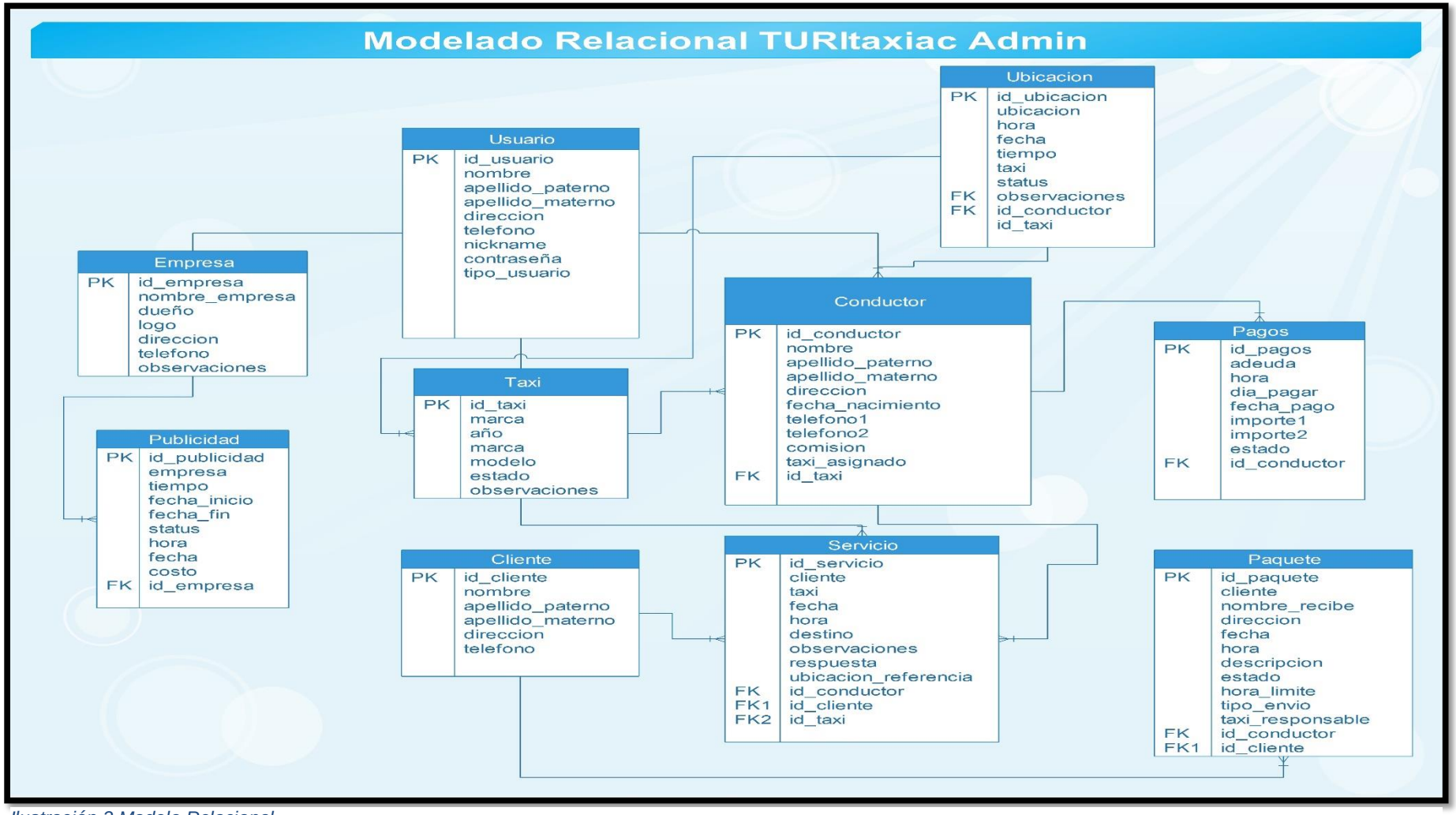

Ilustración 2 Modelo Relacional

9.3 Diagrama de Clases

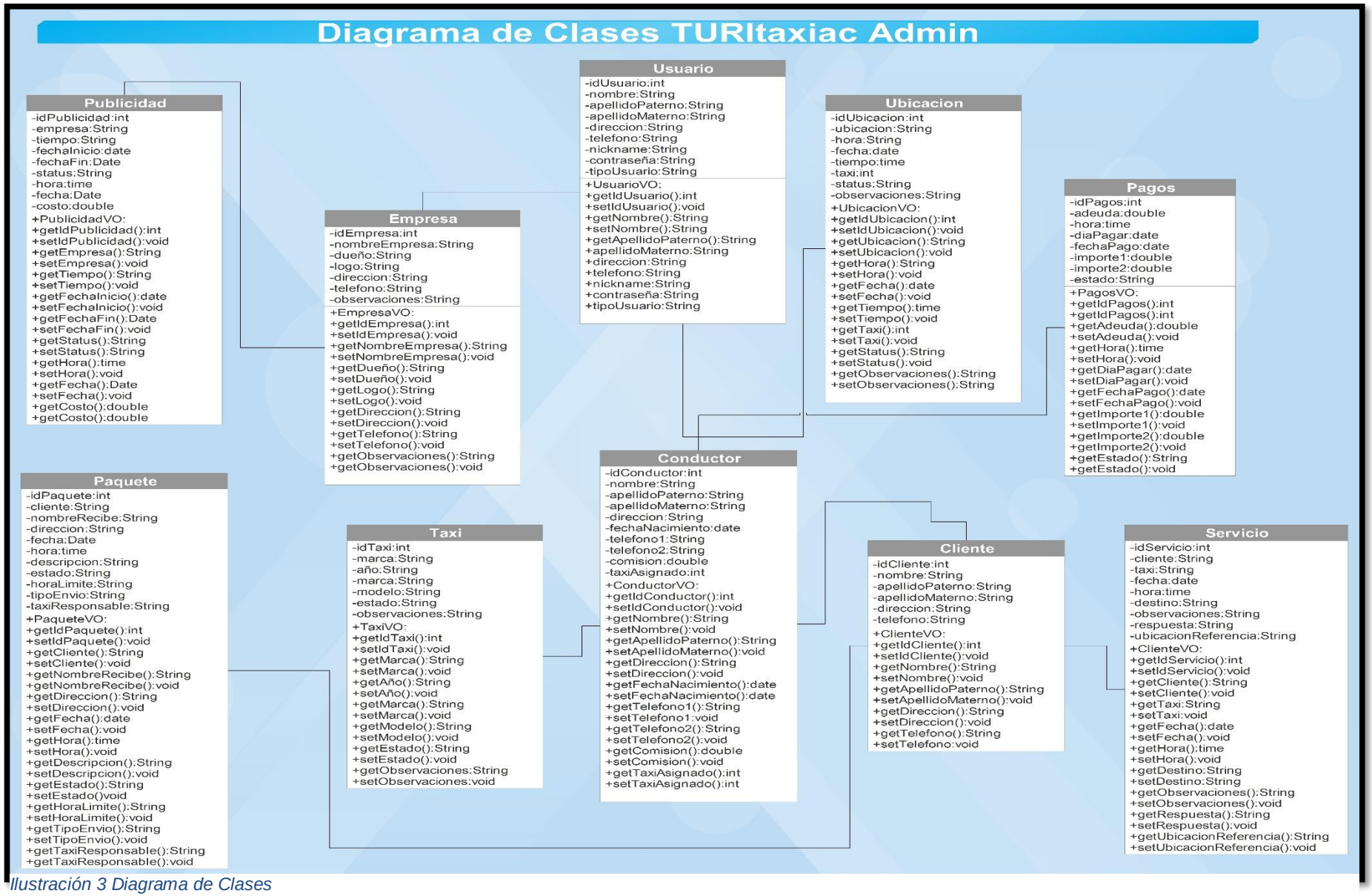

Ilustración 3 Diagrama de Clases

10 Plan De Trabajo

10.1 Cronograma de trabajo

UNIVERSIDAD TECNOLÓGICA DEL CENTRO DE VERACRUZ

Universidad Tecnológica del Centro de Veracruz
(ITI) Ingeniería en Tecnologías de la información
Cronograma de actividades

Nombre del proyecto: TURItaxiac Admin

No	Actividad	Producto (Evidencia de actividad realizada)	P/R	Septiembre-Diciembre	2016 Enero 1	Enero 2	Enero 3	Enero 4	Febrero 1	Febrero 2	Febrero 3	Febrero 4	Marzo 1	Marzo 2	Marzo 3	Marzo 4	Abril 1	Abril 2	Abril 3	Abril 4
1	Protocolo	Protocolo	P																	
			R																	
2	Revisar y pulir en redacción, la problemática, objetivos, justificación, alcance y las limitaciones del proyecto, si así se requiere.	Protocolo	P																	
			R																	
3	Creación de historias de usuario	Historia de usuario	P																	
			R																	
4	Creación del modelo entidad-relación	Diagrama Entidad-Relación	P																	
			R																	
5	Creación diagrama relacional	Diagrama relacional	P																	
			R																	
6	Creación de diagramas de clases	Diagrama de clases	P																	
			R																	
7	Diseño de diccionario de datos	Diccionario de datos	P																	
			R																	
8	Diseño de la base de datos	Base de datos	P																	
			R																	
9	Diseño de las interfaces	Módulos	P																	
			R																	
10	Programación de interfaces	Sistema	P																	
			R																	
11	Implementación	Ejecutable	P																	
			R																	
12	Pruebas funcionales	Documentación de pruebas	P																	
			R																	
13	Creación del glosario	Glosario	P																	
			R																	
14	Creación de manuales	Manuales	P																	
			R																	

Ilustración 4 Cronograma de actividades

11 Historias de Usuario

11.1 Control de acceso de usuarios

Historia de Usuario Turitaxi	
Número: 1	**Usuario:** Todos
Nombre historia: Control de acceso de usuarios	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Baja
Puntos estimados: 3	**Iteración asignada:** 1
Programador responsable: Víctor Gerardo Texcahua Bautista	
Descripción: Antes de que se pueda ingresar al sistema la aplicación deberá solicitar el nombre de usuario y contraseña. Hay dos tipos de usuarios, Administrador e invitado cada uno con distintos permisos de acceso a los menús y acceso a sus funcionalidades que le corresponden.	
Observaciones:	

Tabla 2. Control de acceso de usuarios

11.2 Alta de usuarios

Historia de Usuario Turitaxi	
Número: 2	**Usuario:** Administrador
Nombre historia: Alta de usuarios	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** alto
Puntos estimados: 2	**Iteración asignada:** 1
Programador responsable: Victor Gerardo Texcahua Bautista	
Descripción: El único que podrá dar de alta a los usuarios que tendrán acceso al sistema será el administrador ya que es el único que tendrá acceso al módulo de alta de usuarios, donde deberá ingresar los datos y crear un usuario y contraseña como elegir el tipo de usuario (Administrador o invitado).	
Observaciones:	

Tabla 3. Alta de usuarios

11.3 Alta de unidades

Historia de Usuario Turitaxi	
Número: 3	**Usuario:** Todos
Nombre historia: Alta de unidades	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 1
Programador responsable: Victor Gerardo Texcahua Bautista	
Descripción: Todos los usuarios podrán dar de alta a las unidades (Taxi) con sus respectivos datos, los únicos que podrán eliminar alguna unidad que ya se encuentre dada de alta en el sistema serán los administradores.	
Observaciones: En este mismo modulo se podrá especificar si alguna unidad se encuentra en reparación y así no esté disponible en los demás módulos.	

Tabla 4. Alta de unidades

Alta de conductores

Historia de Usuario Turitaxi	
Número: 4	**Usuario:** Todos
Nombre historia: Alta de conductores	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 1
Programador responsable: Victor Gerardo Texcahua Bautista	
Descripción: Todos los usuarios podrán dar de alta a las conductores con los datos básicos, y así mismo podrán asignarle alguna unidad (taxi) que ya este registrada. Solo el administrador podrá dar de baja a algún conductor.	
Observaciones:	

Tabla 5. Alta de conductores

Historia de Usuario Turitaxi	
Número: 5	**Usuario:** Todos
Nombre historia: Alta de clientes	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 1
Programador responsable: Victor Gerardo Texcahua Bautista	
Descripción: Todos los usuarios podrán dar de alta a los clientes con los datos básicos como poniendo alguna referencia que ayude a ubicar más rápido su dirección también podrán modificar algún cliente que ya se encuentre registrado en el sistema, Solo el administrador podrá dar de baja a algún cliente.	
Observaciones:	

11.4 Alta de clientes

Tabla 6. Alta de clientes

11.5 Alta de empresas

Historia de Usuario Turitaxi	
Número: 6	**Usuario:** Todos
Nombre historia: Alta de empresas	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 1
Programador responsable: Victor Gerardo Texcahua Bautista	
Descripción: Todos los usuarios podrán dar de alta a las empresas que soliciten el servicio de publicidad.	
Observaciones:	

Tabla 7. Alta de empresas

11.6 Servicio de publicidad

Historia de Usuario Turitaxi	
Número: 7	**Usuario:** Todos
Nombre historia: Servicio de publicidad	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 1
Programador responsable: Victor Gerardo Texcahua Bautista	
Descripción: En este módulo todos los usuarios podrán dar de alta el servicio de publicidad a una empresa ya registrada, en este módulo podrán ingresar el inicio y fin del servicio como el costo del servicio.	
Observaciones: para poder brindar el servicio de publicidad a una empresa, la empresa ya debe estar dada de alta.	

Tabla 8. Servicio de publicidad

11.7 Pagos de comisión

Historia de Usuario Turitaxi	
Número: 8	**Usuario:** Todos
Nombre historia: Pagos de comisión	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 1
Programador responsable: Victor Gerardo Texcahua Bautista	
Descripción: En este módulo, el sistema generara una comisión a cada conductor diariamente los usuarios registraran el pago de cada conductor.	
Observaciones:	

Tabla 9. Pagos de comisión

11.8 Control de ubicación

Historia de Usuario Turitaxi	
Número: 9	**Usuario:** Todos
Nombre historia: Control de ubicación	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 3	**Iteración asignada:** 1
Programador responsable: Victor Gerardo Texcahua Bautista	
Descripción: En este módulo se llevara el control de la ubicación de cada unidad, especificando si están brindando un servicio o se encuentran disponibles, se registrara en qué lugar se encuentran o hacia donde se dirigen.	
Observaciones: para poder llevar el control se basaran en la comunicación que se tiene por radio con los conductores.	

Tabla 10. Control de ubicación

11.9 Control de servicios

Historia de Usuario Turitaxi	
Número: 10	**Usuario:** Todos
Nombre historia: Control de servicios	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 3	**Iteración asignada:** 1
Programador responsable: Victor Gerardo Texcahua Bautista	
Descripción: En este módulo se llevara el control de todas las unidades en servicio para así poder asignarles algún servicio que sea solicitado por algún cliente, este módulo manejara una ruleta donde el servicio se asignara por turnos.	
Observaciones: En la ruleta donde se asignan turnos solo estarán las unidades que hayan pagado su comisión.	

Tabla 11. Control de servicios

11.10 Control de servicio de paquetería

Historia de Usuario Turitaxi	
Número: 11	**Usuario:** Todos
Nombre historia: Control de servicio de paquetería	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Victor Gerardo Texcahua Bautista	
Descripción: En este módulo se llevara el control de los servicios de paquetería, se podrá dar de alta algún servicio, y se le podrá dar seguimiento hasta que el paquete este en su destino.	
Observaciones:	

Tabla 12. Control de servicio de paquetería

11.11 Actividades

1.- Protocolo

En esta actividad se llevará a cabo una investigación y análisis exhaustivo de la problemática de la empresa TuritaxiAC, y así poder realizar una buena planeación de la metodología, procesos y faces que se llevaran a cabo durante el desarrollo del proyecto

2.- Revisión de protocolo

Se realizara una revisión del protocolo para corregir puntos ya retroalimentados, y definir qué puntos se agregaran al documento.

3.- Creación de historias de usuario

Se levantaran los requerimientos del sistema de acuerdo al lenguaje del cliente para después poder interpretarlos para un buen diseño del sistema.

4.- Creación de diagramas Entidad-Relación

Se realizará el diagrama E-R para identificar las relaciones de cada tabla y así poder tener un mejor modelado del sistema, y posteriormente basarse de este diagrama para realizar el diagrama relacional y de clases.

5.- Creación diagrama relacional

Ya contando con el diagrama E-R se procederá a realizar el diagrama relacional para conocer la relación que tendrá cada tabla (PK, FK) y así poder tener un buen diseño de la base de datos.

6.- Creación diagrama de clases

Se estructurara el diagrama de clases partiendo del diagrama entidad-relación, para así ir asignando a cada atributo el tipo de dato correspondiente como también si será público, privado, protegido etc.

7.- Diseño del diccionario de datos

Después de haber creado el diagrama de clases se procederá a realizar el diccionario de datos para asignar a qué tipo de dato pertenecerá como la cantidad de caracteres, para así poder almacenarlo en la base de datos y tenga un correcto funcionamiento.

8.- Diseño de la base de datos

Ya contando con el diccionario de datos el siguiente paso en el desarrollo del proyecto será la creación de la base de datos por medio del gestor HeidiSQL para así poder empezar a almacenar los datos para el funcionamiento del sistema.

9.- Diseño de las interfaces

Haciendo uso de la herramienta JavaFx y de las hojas de estilo (CSS) se realizara el diseño de las interfaces o módulos del sistema para más adelante poder darles funcionalidad.

10.- Programación de las interfaces

En este punto es el que tomará más tiempo en el desarrollo del proyecto ya que se inicia la codificación en lenguaje Java de los módulos para darles funcionalidad apoyándose del IDE Eclipse.

11.- Implementación

Ya teniendo codificado el sistema y con el ejecutable se procederá a la instalación para ponerlo en funcionamiento y realizarle las pruebas pertinentes

12.- Pruebas funcionales

En este punto el Tester encargado del proyecto realizará las pruebas adecuadas al sistema para encontrar errores y así poder corregirlos a tiempo para el correcto funcionamiento del sistema.

13.- Creación del glosario

Se realizará el glosario con términos y abreviaturas que se manejaron durante el protocolo y desarrollo de todo el proyecto.

14.- Creación de manuales

Por ultimo una vez implementado el sistema se deberán entregar los manuales de usuario al cliente para que conozca el correcto uso del sistema y en cualquier caso necesario se pueda apoyar del manual.

11.12 Roles y funciones

Rol	Encargado	Función
Cliente	Arturo Merino Castillo	• Coordinar los esfuerzos generales del proyecto, ayudando a cada uno de sus integrantes a cumplir sus objetivos particulares. Al final, se cumplirá el objetivo general. • Observar cada actividad para detectar y resolver inconvenientes. Lograr el mejor uso de los recursos disponibles.
Programador	Victor Gerardo Texcahua Bautista	• Convertir la especificación del sistema en código fuente ejecutable utilizando uno o más lenguajes de programación, así como herramientas de software de apoyo a la programación. • Explorar los diferentes ambientes en que el sistema puede ser desarrollado Reunirse con otros miembros del equipo de programadores
Tester	Dora Luz Bautista Amador	• Encargarse de asegurar la calidad de cada uno de los productos (documentos, prototipos, etc.). • Participación en el proceso de especificación del Sistema • Interacción con el diseñador • Realizar los tests, apoyado por los programadores. Informar sobre los resultados obtenidos.
Consultor	Asesor	• Es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto. • Guía al equipo para resolver un problema específico.
Gestor (*Big boss*)	Asesor	• Es el vínculo entre clientes y programadores, ayuda a que el equipo trabaje efectivamente creando las condiciones adecuadas. • Su labor esencial es de coordinación.

Tabla 13 Roles y funciones

12 Anexos

12.1 Ventana Inicio de Sesión

Ilustración 5 Inicio de sesion

12.2 Ventana Principal

Ilustración 6 Ventana principal

12.3 Ventana Alta Usuarios

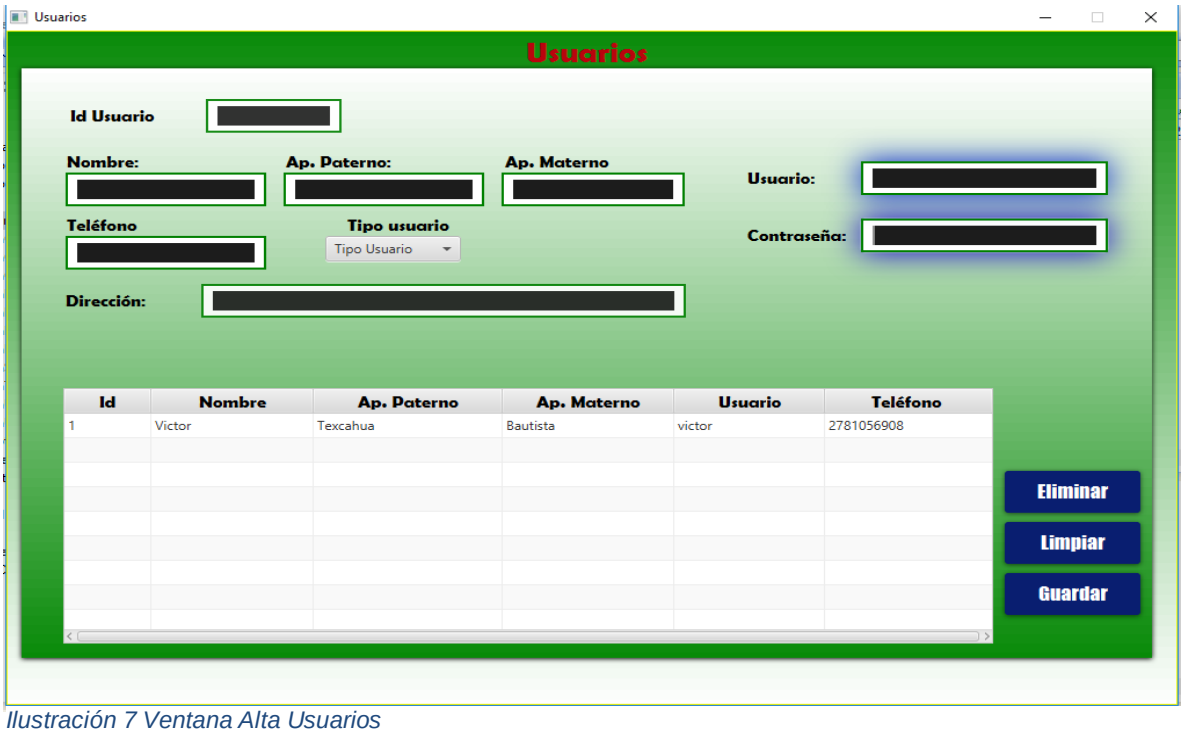

Ilustración 7 Ventana Alta Usuarios

12.4 Ventana Alta de clientes

Ilustración 8 Alta Clientes

12.5 Ventana alta taxis

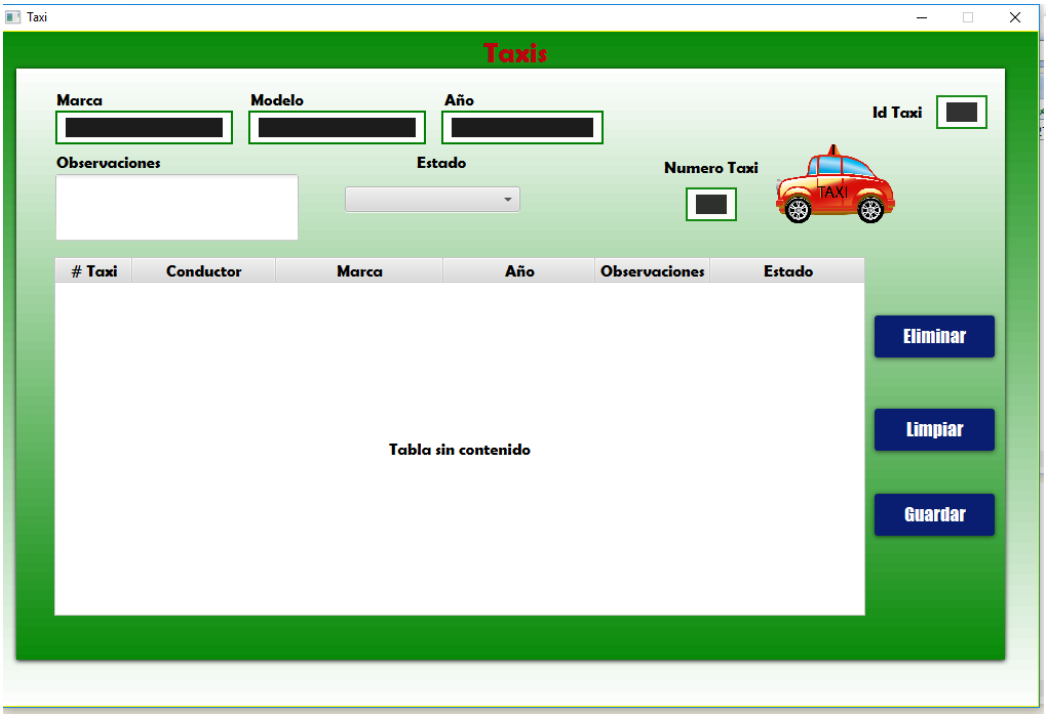

Ilustración 9 Alta taxis

12.6 Ventana alta conductores

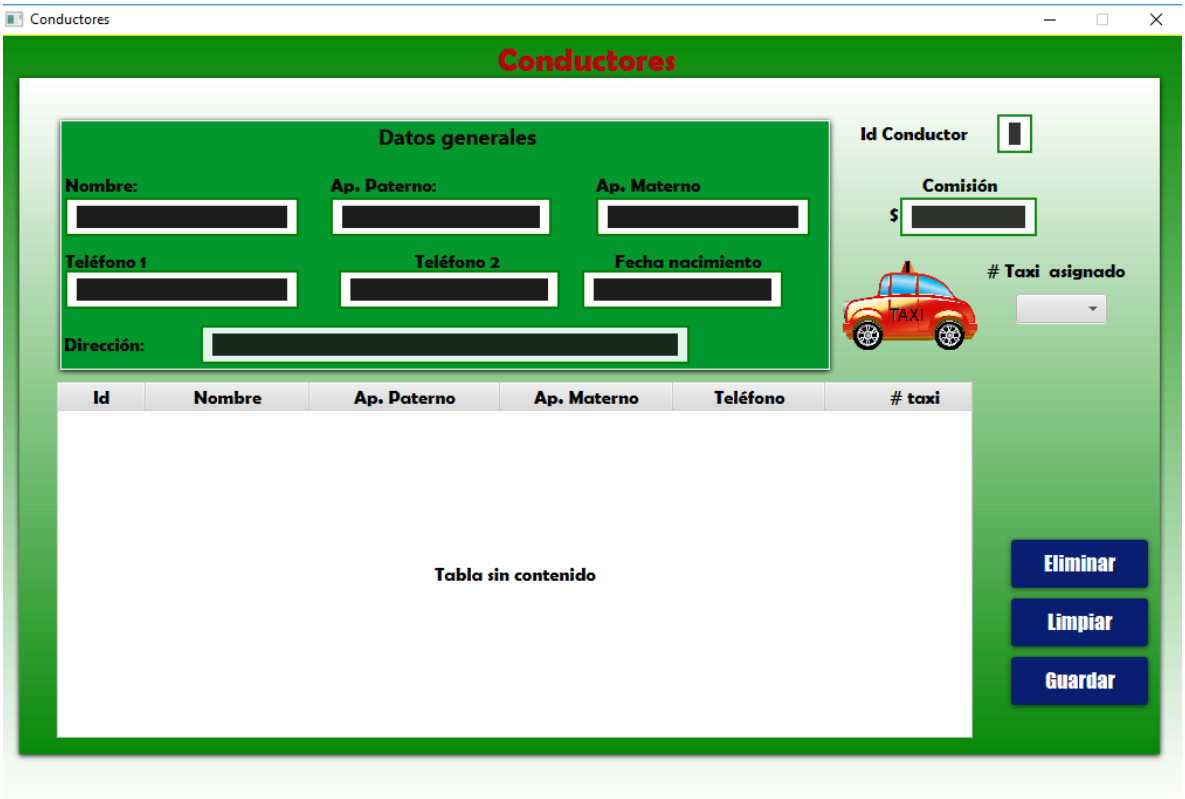

Ilustración 10 Alta conductores

12.7 Ventana alta empresa

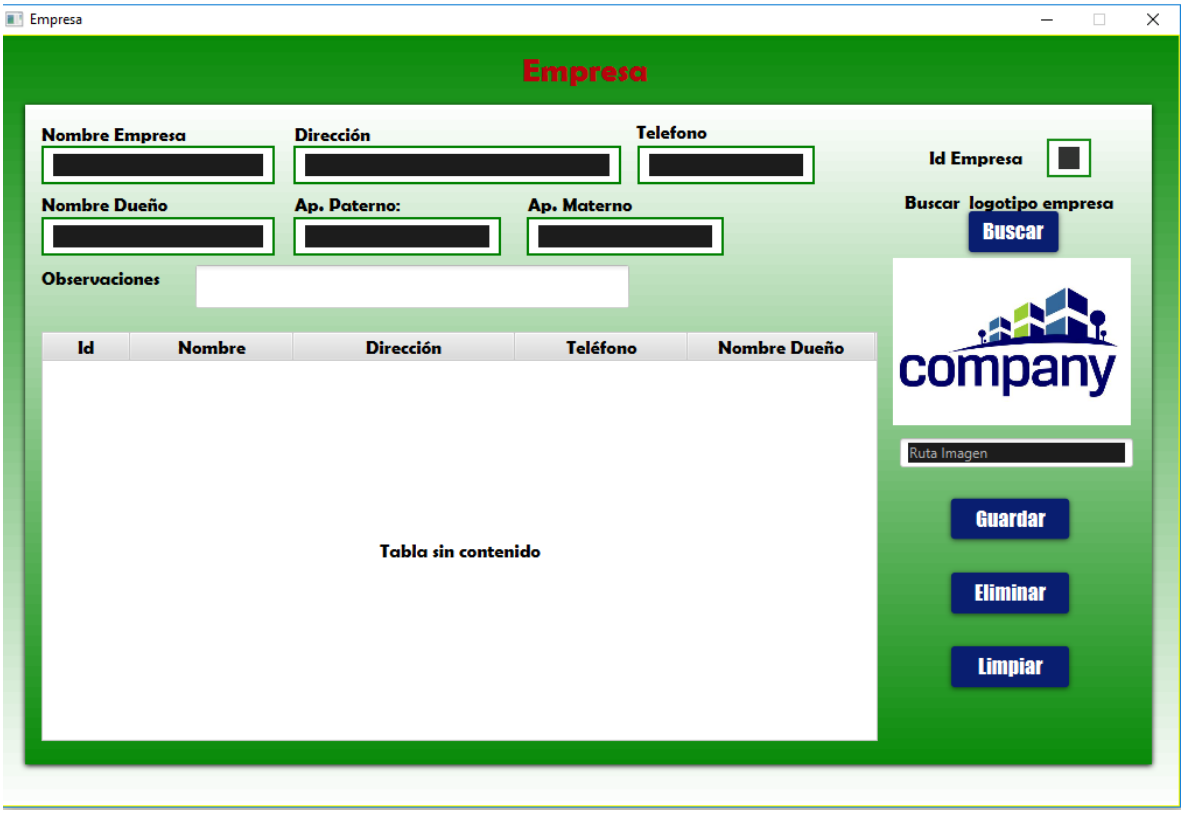

Ilustración 12 Alta empresa

12.8 Ventana Ubicación

Ilustración 11 Ubicación

12.9 Ventana paquetería

Ilustración 13 Paquetería

12.10 Ventana servicios

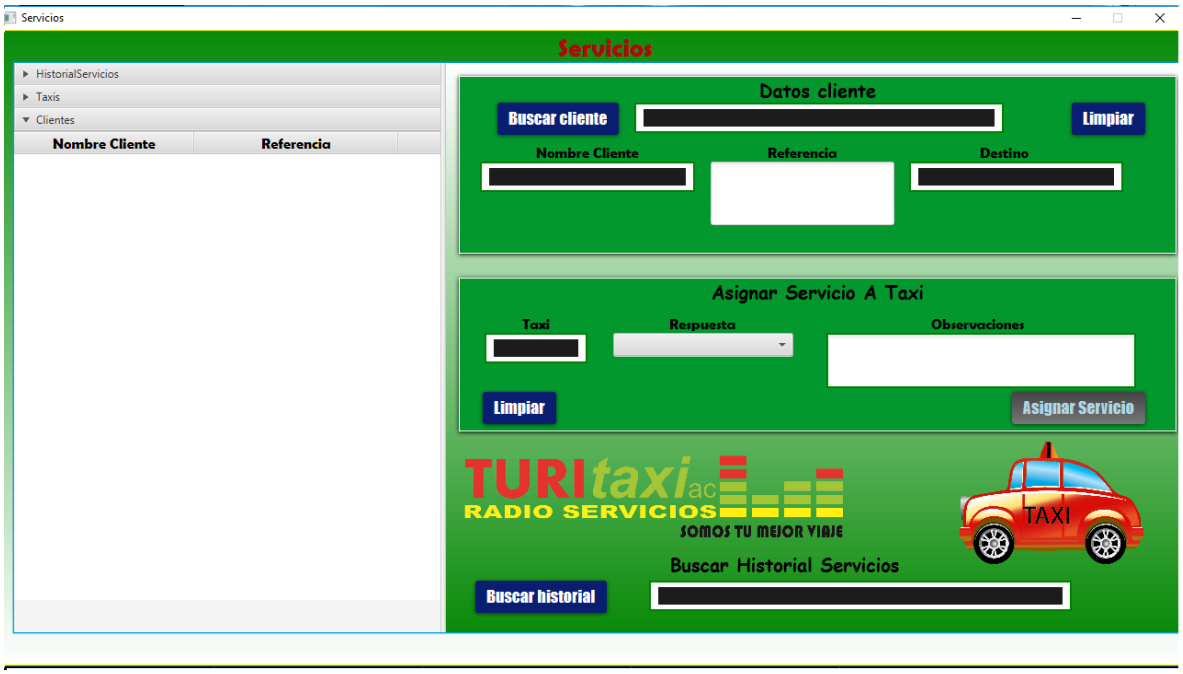

Ilustración 14 Servicios

12.11 Ventana de pagos

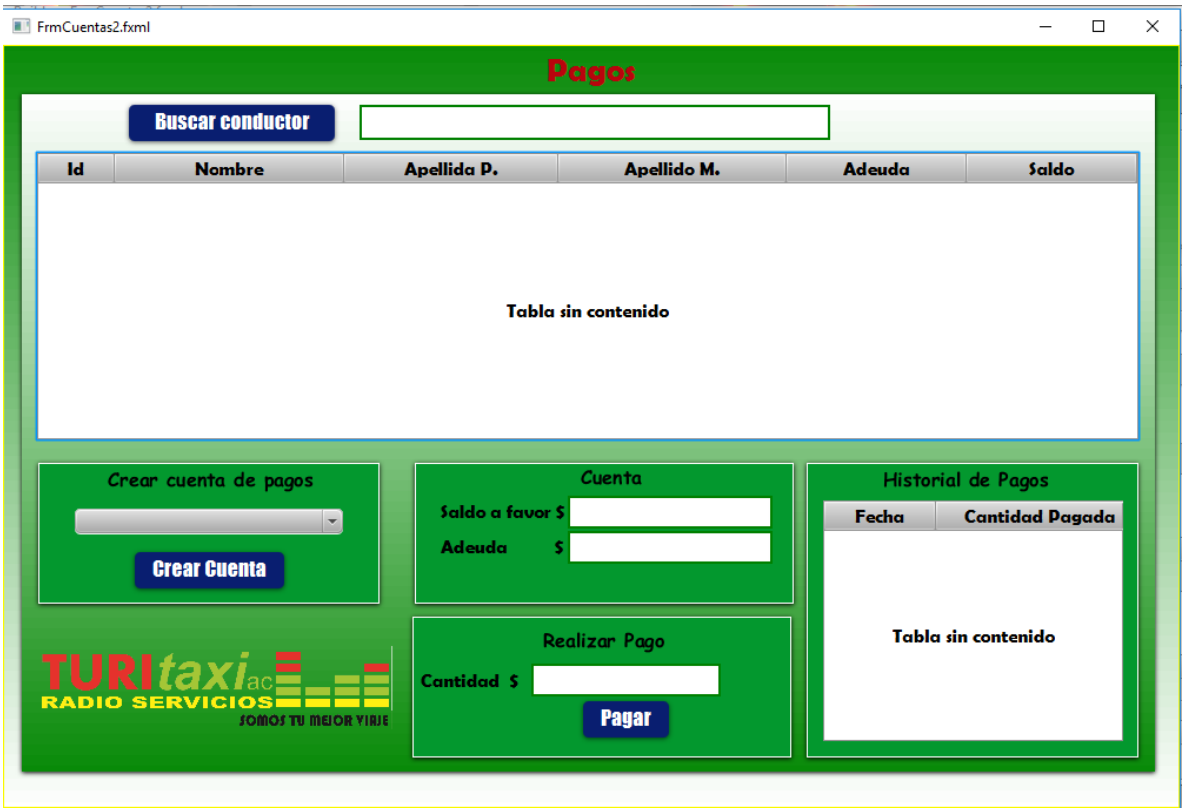

Ilustración 15 Pagos

13 Bibliografía

Alexander. (09 de Julio de 2011). *IntroduccionJavaFx*. Obtenido de http://introduccionajavafx.blogspot.com/2011/07/introduccion-javafx.html

Bravo, I. M. (04 de Noviembre de 2006). *Informatica.* Obtenido de http://indira-informatica.blogspot.mx/2007/09/qu-es-mysql.html

Bustamante Dayana, R. J. (01 de Marzo de 2014). *UNIVERSIDAD NACIONAL EXPERIMENTAL DE LOS LLANOS.* Obtenido de http://blogs.unellez.edu.ve/dsilva/files/2014/07/Metodologia-XP.pdf

Julián Pérez Porto, A. G. (01 de Enero de 2010). *DEFINICION.DE*. Obtenido de http://definicion.de/java/

Maria. (19 de Septiembre de 2007). *Empresa y economia.republica*. Obtenido de http://empresayeconomia.republica.com/aplicaciones-para-empresas/heidisql-la-gestion-de-tus-bases-de-datos-a-golpe-de-clic.html

Patricio Letelier, M. C. (05 de Enero de 2006). Obtenido de Cyta: http://www.cyta.com.ar/ta0502/v5n2a1.htm