Contenido

[1. Introducción 3](#_Toc524907853)

[2. Antecedentes 4](#_Toc524907854)

[2.1. Uber 4](#_Toc524907855)

[2.2. Cabify 5](#_Toc524907856)

[2.3. Easy Taxi 6](#_Toc524907857)

[3. Justificación 7](#_Toc524907858)

[4. Descripción del problema 7](#_Toc524907859)

[5. Formulación del problema 8](#_Toc524907860)

[6. Objetivos 9](#_Toc524907861)

[6.1. Objetivo general 9](#_Toc524907862)

[6.2. Objetivos específicos 9](#_Toc524907863)

[7. Alcance 10](#_Toc524907864)

[7.1. Alcance web 10](#_Toc524907865)

[7.2. Alcance móvil 10](#_Toc524907866)

[8. Técnicas orientadas a mejorar la productividad 10](#_Toc524907867)

[8.1. Uso de una metodología ágil de desarrollo de software 10](#_Toc524907868)

[8.2. Uso de herramientas durante el ciclo de vida 10](#_Toc524907869)

[8.3. Desarrollo de software basado en componentes 10](#_Toc524907870)

[9. Actividades adicionales para asegurar la calidad 11](#_Toc524907871)

[10. Innovación en el proyecto 11](#_Toc524907872)

[10.1. Funcional 11](#_Toc524907873)

[10.2. No funcional 12](#_Toc524907874)

[10.2.1. Aplicación web progresiva 12](#_Toc524907875)

[10.2.2. Aplicaciones móviles multiplataforma nativas 13](#_Toc524907876)

[10.3. Herramienta 14](#_Toc524907877)

[11. Arquitectura del software 15](#_Toc524907878)

[12. Ingeniería inversa y re-ingeniería 17](#_Toc524907879)

[12.1. Ingeniería inversa 17](#_Toc524907880)

[12.2. Re-ingeniería 18](#_Toc524907881)

[13. Descripción de herramientas de desarrollo 20](#_Toc524907882)

[13.1. PhpStorm 20](#_Toc524907883)

[13.2. Apache 20](#_Toc524907884)

[14.3. Enterprise Architect 21](#_Toc524907885)

[13.4. Git 21](#_Toc524907886)

[13.5. Visual Studio Code 22](#_Toc524907887)

[13.6. Postman 23](#_Toc524907888)

[14. Proceso del software 23](#_Toc524907889)

[14.1. Modelo de requisito 23](#_Toc524907890)

[14.2. Modelo de arquitectura del software a nivel del código 23](#_Toc524907891)

[14.3. Modelo de arquitectura del software a nivel de despliegue 23](#_Toc524907892)

[14.4. Modelo de persistencia en el servidor y móvil 24](#_Toc524907893)

[16. Referencias 24](#_Toc524907894)

# 1. Introducción

Un factor importante para el crecimiento de cualquier institución académica es la movilidad, ya que resulta indispensable para los ciudadanos de la misma (plantel docente, estudiantil y administrativo) disminuir los tiempos de deslazamiento; es por ello que el uso de taxis, como servicio público, se ha incrementado en los últimos años de forma considerable. Esto debido a la comodidad y rapidez que ofrece el servicio hacia sus usuarios.

A medida que ha aumentado la demanda de este servicio, también se han incrementado el número de usuarios inconformes debido a el cobro elevado de la tarifa por parte de los taxistas, también por la poca seguridad ofrecida ya que en la ciudad existen muchos casos de robos, secuestros y maltrato durante el uso de este servicio.

Con el siguiente proyecto se propone la creación de un software para el control y uso de un servicio de transporte universitario realizado únicamente por y para los ciudadanos de la institución, pudiendo ellos ofrecer transporte a sus conciudadanos, haciendo un bien social y al mismo tiempo generando una pequeña fuente de ingreso o también pudiendo ellos recibir un servicio de transporte cómodo, seguro y eficiente.

# 2. Antecedentes

## 2.1. Uber



Uber es una plataforma tecnológica que pone en contacto a socios conductores con pasajeros gracias a una aplicación móvil.

Funciona de la siguiente manera. Primero, solicitas un viaje desde la aplicación en cualquier ciudad donde Uber esté activo. Tu solicitud se enviará a los socios conductores más cercanos. Cuando un conductor acepte tu solicitud de viaje, podrás consultar en la aplicación el tiempo estimado de llegada de tu conductor a la ubicación de recogida. También te llegará una notificación cuando el conductor esté a punto de llegar.

Al solicitar un viaje, la aplicación te proporciona el nombre del conductor, el tipo de vehículo y su número de matrícula para que puedas reconocerlo sin problemas. Cuando estés dentro del vehículo, comprueba que el destino que has introducido en la aplicación sea el correcto. Los conductores utilizan su aplicación Uber para confirmar los detalles del viaje. Si prefieres alguna ruta en concreto, díselo al conductor.

Cuando llegues a tu destino, el conductor finalizará el viaje. El precio del viaje se calcula automáticamente y se te cobra a través del método de pago vinculado a tu cuenta de Uber. En determinadas ciudades Uber te permitirá pagar con dinero en efectivo.

Inmediatamente después, la aplicación te pedirá que valores la experiencia de tu viaje. A los conductores también se les pide que valoren el viaje.

## 2.2. Cabify



Al igual que en Uber, el pago se realiza online y no hay necesidad de ningún intercambio de dinero físico entre el conductor y el pasajero. Estas son las pautas para disfrutar de sus servicios:

1. Darse de alta en la web de la plataforma y, muy recomendable, descargarse la aplicación en el móvil.
2. Rellenar los datos de registro: nombre, email y número de teléfono.
3. Insertar la dirección de destino: tras ello, el usuario conoce quién será su chófer, en cuánto tiempo le recoge y qué tipo de coche será.
4. Se puede reservar el vehículo (sin coste) en caso de requerir sus servicios para una determinada fecha y hora. La cancelación es gratuita siempre y cuando el coche no haya llegado a la zona acordada.
5. Elegir el tipo de vehículo. Lea las indicaciones bajo el apartado 'Tipos de coches de Cabify'.
6. Seleccionar un método de pago: o PayPal o tarjeta. Cabify cobra por distancia y no por tiempo.

## 2.3. Easy Taxi

Easy Taxi logo.png

Fundada en San Pablo (Brasil) en junio de 2011, llegó a la Argentina a comienzos del año pasado y empezó a operar unos meses después. Para usarla, se debe descargar la aplicación en forma gratuita y crear una cuenta por única vez con nombre y contraseña. De esta manera, el GPS del celular detectará automáticamente la calle en la que se encuentra el pasajero y éste sólo deberá ingresar la altura para facilitar su búsqueda.

Una vez ubicado exactamente en el mapa, se presiona el botón verde “Pedir taxi". A partir de ese momento, la aplicación realiza una búsqueda basándose en el taxista que está más cercano a ese punto. Apenas lo encuentra, muestra en qué lugar del mapa está con un símbolo de un vehículo. Si el taxista acepta el viaje, aparecerá una notificación en la pantalla y el usuario recibirá automáticamente toda la información del conductor.

Así, verá en su pantalla el nombre completo del conductor, foto, número de celular, modelo del auto y patente. Además, puede seguir en tiempo real el recorrido que el taxi hace hasta el domicilio desde el cual se pidió el servicio. También el taxista recibirá los datos del usuario, como su nombre y número de teléfono. En caso de querer suspender el viaje se puede optar por el botón "Cancelar" y elegir el motivo de cancelación.

# 3. Justificación

Debido al reciente cambio de la ciudad universitaria a lejanías de la ciudad la demanda de un servicio de transporte eficiente, eficaz y seguro ha incrementado de forma considerable. Ya que en la institución se cuenta con miembros de distintas clases sociales no se puede asumir que todos poseen un vehículo o los recursos necesarios para tomar un servicio público (taxi) diariamente, debido a sus altos costos, por lo que el transportarse de un lado a otro se ha vuelto un desafío financiero, además hay que recordar que en este servicio se puede sufrir todo tipos de peligros como ser robos, secuestros y abusos.

Por tal motivo la introducción de una aplicación capaz de poder ayudar a los miembros de la universidad a generar recursos y al mismo tiempo transportarse unos a otros de forma rápida, sencilla y segura viene a ser la mejor solución.

# 4. Descripción del problema

* Consumo excesivo de tiempo

Con la mudanza de los previos universitarios fuera de la ciudad el uso de transporte público (micros) ha sido una de las opciones como medio de desplazamiento para los ciudadanos universitarios. Pero como ellos tiene una ruta fija que abarca grandes recorridos dentro de la ciudad y un constante alojo y desalojo de los pasajeros ha vuelto que un viaje promedio dure hasta más de 45 minutos perjudicando de esta forma a todo aquel que use este servicio.

* Altos precios del pasaje en taxi

Como la distancia hasta las nuevas instalaciones de la universidad es considerablemente larga los precios de una carrera de taxi son demasiado altos superando fácilmente los 30 bs, por lo cual hacer uso de este servicio diariamente significaría un golpe económico muy grande en los recursos de cualquier miembro de la universidad.

* Inseguridad

Ya sea haciendo uso de un medio de transporte público masivo (microbuses) o de los taxis la población universitaria se ha envuelto en todo tipo de peligros siendo los principales los robos, asaltos, secuestros y los abusos.

* Falta de espacio de aparcamiento

Debido a la gran cantidad de miembros que posee la universidad seria casi impensable que todos y cada uno de ellos puedan disponer de un parqueo para sus propios autos particulares, por lo que al llevar uno su carro corre el riesgo de no encontrar un lugar donde aparcar o perder tiempo en el proceso.

# 5. Formulación del problema

Se pretende desarrollar una aplicación para el uso del servicio de transporte universitario, el cual permitirá a todos los miembros de la universidad registrarse para poder prestar servicio como conductores o recibir el servicio como pasajero. Dicha aplicación estará separada en una versión web y otra móvil, siendo la versión web para el registro del usuario, sus vehículos, las rutas que realizará y la visualización de los movimientos realizados de su cuenta monetaria. Y la parte móvil para la solicitud del servicio, señalando el punto de encuentro dentro de la ruta especificada por el conductor, recibiendo notificaciones según la proximidad de ambos, además dicha aplicación móvil debe adaptarse a distintos sistemas operativos siendo los más comunes Android y IOS.

# 6. Objetivos

## 6.1. Objetivo general

Desarrollar una aplicación web y móvil para el servicio de transporte universitario, el cual será ofrecido únicamente por y para la población universitaria, siendo estos los estudiantes, docentes y personal administrativo.

## 6.2. Objetivos específicos

* Diseñar e implementar una aplicación web progresiva, la cual permita a los usuarios poder registrarse y hacer uso del servicio.
* Diseñar e implementar una aplicación móvil multiplataforma basados en sistemas Android y IOS que permita solicitar el servicio, y además muestre el progreso del mismo a través de la geolocalización.
* Ofrecer a la población universitaria una herramienta el cual proporcionará una alternativa económica, rápida, sencilla y segura para su movilización hacia los previos de la Universidad.
* Liberar el congestionamiento automovilístico en los aparcamientos de la universidad.
* Ofrecer una pequeña fuente de ingreso a aquellos que deseen prestar servicio como conductor para el recojo de pasajeros mientras se encuentra camino hacia la universidad.

# 7. Alcance

## 7.1. Alcance web

## 7.2. Alcance móvil

# 8. Técnicas orientadas a mejorar la productividad

## 8.1. Uso de una metodología ágil de desarrollo de software

El desarrollo ágil de software envuelve un enfoque para la toma de decisiones en los proyectos de software, que se refiere a métodos de ingeniería del software basados en el desarrollo iterativo e incremental, donde los requisitos y soluciones evolucionan con el tiempo según la necesidad del proyecto. Así el trabajo es realizado mediante la colaboración de equipos auto-organizados y multidisciplinarios, inmersos en un proceso compartido de toma de decisiones a corto plazo.

## 8.2. Uso de herramientas durante el ciclo de vida

Hoy en día se encuentra una vasta cantidad de tecnologías y herramientas que facilitan y agilizan el desarrollo de software, ayudando en todo el ciclo de vida de un proyecto, desde el diseño y modelado hasta la implementación, puesta en marcha y prueba. Ayudando de manera significativa a la reducción de tiempo y proporcionando agilidad en el proceso de desarrollo.

## 8.3. Desarrollo de software basado en componentes

La complejidad de los sistemas computacionales actuales nos ha llevado a buscar la reutilización del software existente. El desarrollo de software basado en componentes permite reutilizar piezas de código pre-elaborado que permiten realizar diversas tareas, conllevando a diversos beneficios como las mejoras a la calidad, la reducción del ciclo de desarrollo y el mayor retorno sobre la inversión.

El uso de este paradigma posee algunas ventajas:

* Reutilización del software.
* Simplifica las pruebas.
* Simplifica el mantenimiento del sistema.
* Mayor calidad

# 9. Actividades adicionales para asegurar la calidad

Se realizó una gran cantidad de distintas pruebas verificando en todos los posibles ambientes imaginados verificando de esta manera que se están cumpliendo con todos los requisitos solicitados.

Se cuenta con un servicio de hosting estable y seguro, siendo que nuestra aplicación web y móvil funcionará únicamente con conexión a internet es muy necesario que esta se encuentre alojada en un lugar confiable, seguro y libre de caídas, ya que un problema en el servidor ocasionaría fallas al funcionamiento de la aplicación.

Se trabajó con una considerable población de datos para la verificación de la velocidad de respuesta del servicio.

# 10. Innovación en el proyecto

## 10.1. Funcional

Se agregó una capa de seguridad a través del reconocimiento facial. Utilizando el api de Kairos. El proceso consta de cuatro módulos principales:

1. Detección de la cara: detecta que hay una cara en la imagen, sin identificarla. Si se trata de un video, también podemos hacer un seguimiento de la cara. Proporciona la localización y la escala a la que encontramos la cara.
2. Alineación de la cara: localiza las componentes de la cara y, mediante transformaciones geométricas, la normaliza respecto propiedades geométricas, como el tamaño y la pose, y fotométricas, como la iluminación. Para normalizar las imágenes de caras, se pueden seguir diferentes reglas, como la distancia entre las pupilas, la posición de la nariz, o la distancia entre las comisuras de los labios. También se debe definir el tamaño de las imágenes y la gama de colores. Normalmente, para disminuir la carga computacional del sistema, se acostumbra a utilizar imágenes pequeñas en escala de grises. A veces también se realiza una ecualización del histograma.
3. Extracción de características: proporciona información para distinguir entre las caras de diferentes personas según variaciones geométricas o fotométricas.
4. Reconocimiento: el vector de características extraído se compara con los vectores de características extraídos de las caras de la base de datos. Si encuentra uno con un porcentaje elevado de similitud, nos devuelve la identidad de la cara; si no, nos indica que es una cara desconocida.

## 10.2. No funcional

### 10.2.1. Aplicación web progresiva

Las Progressive Web Apps son experiencias que combinan lo mejor de la Web y lo mejor de las apps. Están disponibles para los usuarios a partir de la primera visita en una pestaña del navegador y no requieren instalación. A medida que el usuario compila progresivamente una relación con la app con el paso del tiempo, se hace más y más poderosa. Se carga rápidamente, incluso con redes débiles, envía notificaciones push relevantes, tiene un ícono en la pantalla principal y se carga como experiencia de pantalla completa y de primer nivel.

¿Qué es una Progressive Web App?

Una Progressive Web App es:

* Progresiva: funciona para todos los usuarios, sin importar la elección de navegador, porque está construida con mejora progresiva como principio central.
* Adaptable: se adapta a cualquier factor de formulario, sea escritorio, móvil, tablet o lo que venga en el futuro.
* Independiente de la conectividad: mejorada con service workers para trabajar sin conexión o con redes de mala calidad.
* Estilo app: al usuario le parece una app con interacciones y navegación estilo app, porque está construida con modelo de shell de app.
* Fresca: siempre actualizada gracias al proceso de actualización de service worker.
* Segura: emitida vía HTTPS para evitar intromisiones y para garantizar que el contenido no se haya manipulado.
* Descubrible: se puede identificar como "app" gracias al manifiesto W3C y al alcance de registro de service worker, lo que permite que los motores de búsqueda la encuentren.
* Posibilidad de volver a interactuar: facilita la posibilidad de volver a interactuar a través de funciones como notificaciones push.
* Instalable: les permite a los usuarios "conservar" las apps que les resultan más útiles en su pantalla principal sin la molestia de una tienda de app.
* Vinculable: se puede compartir fácilmente vía URL, no requiere instalación compleja.

### 10.2.2. Aplicaciones móviles multiplataforma nativas

Los desarrolladores front normalmente trabajamos dentro del ecosistema de los navegadores web. Por lo general, el desarrollo de aplicaciones móviles nativas ha sido un mundo aparte para nosotros.

Pero parece que los tiempos están cambiando y, con la promesa de poder programar aplicaciones tanto para iOS como para Android con un solo lenguaje de programación (en este caso Javascript), han nacido los llamados Cross-Platforms frameworks, como Ionic, NativeScript y React Native, de modo que los frontend que queremos matar el gusanillo de hacer apps nativas con JS ya lo tenemos más fácil.

Los marcos multiplataforma proporcionan a los desarrolladores un conjunto completo de herramientas diseñadas para aumentar la productividad al resolver problemas comunes. La pregunta es qué marco es el mejor para usted. Para ayudarlo a responder esta pregunta, hemos preparado esta lista de marcos multiplataforma para el desarrollo móvil.

Con casi 3 millones de aplicaciones en Google Play, el sistema operativo Android domina el panorama móvil. Las personas, las pequeñas empresas y las grandes empresas trabajan arduamente para establecer una fuerte presencia móvil y obtener su parte del mercado. Sin embargo, no todos tienen la experiencia y los recursos necesarios para crear una aplicación desde cero utilizando herramientas nativas.

## 10.3. Herramienta

Uso de React Native. React Native es un framework que permite a los desarrolladores implementar apps nativas para dispositivos móviles utilizando Javascript. De momento los sistemas operativos mínimos soportados son: Android 4.1 (API 16) y >= iOS 8.0.

Las RN apps tienen acceso directo a todas la APIs y views nativas que ofrecen los sistemas operativos nativos. De forma que la experiencia de usuario y el rendimiento es el mismo de una aplicación nativa.

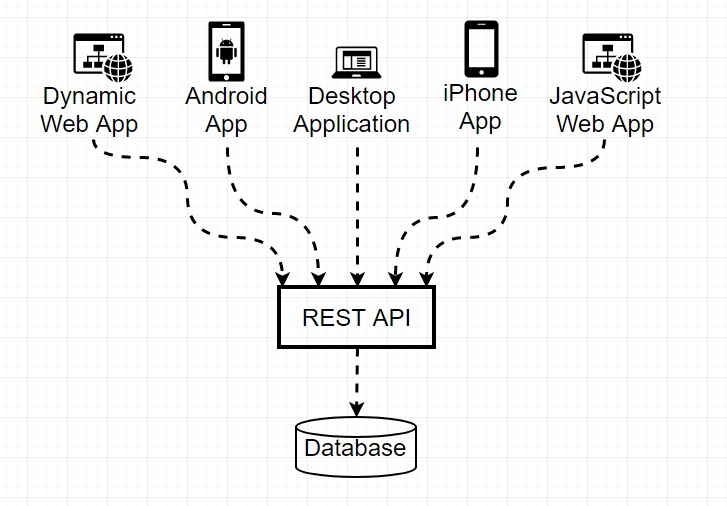
Lo primero que se podría suponer es que React Native compila código JS en el correspondiente código nativo directamente. Pero esa empresa es bastante dura de realizar, ya que Java y Objective C/Swift son lenguajes fuertemente tipados mientras que Javascript no lo es.

En lugar de eso, RN hace algo más inteligente: React Native es, en esencia, un conjunto de componentes React, donde cada uno de ellos tiene su correspondiente equivalente en views y componentes nativos.

# 11. Arquitectura del software

Arquitectura de software. La arquitectura de software es un conjunto de patrones que proporcionan un marco de referencia necesario para guiar la construcción de un software, permitiendo a los programadores, analistas y todo el conjunto de desarrolladores del software compartir una misma línea de trabajo y cubrir todos los objetivos y restricciones de la aplicación. Es considerada el nivel más alto en el diseño de la arquitectura de un sistema puesto que establecen la estructura, funcionamiento e interacción entre las partes del software.

La arquitectura de software forma la columna vertebral para construir un sistema de software, es en gran medida responsable de permitir o no ciertos atributos de calidad del sistema entre los que se destacan la confiabilidad y el rendimiento del software. Además, es un modelo abstracto reutilizable que puede transferirse de un sistema a otro y que representa un medio de comunicación y discusión entre participantes del proyecto, permitiendo así la interacción e intercambio entre los desarrolladores con el objetivo final de establecer el intercambio de conocimientos y puntos de vista entre ellos.



El término REST (Representational State Transfer) un servicio REST es un conjunto de restricciones con las que podemos crear un estilo de arquitectura software, la cual podremos usar para crear aplicaciones web respetando HTTP.

Hoy en día la mayoría de las empresas utilizan API REST para crear servicios. Esto se debe a que es un estándar lógico y eficiente para la creación de servicios web.

Según Fielding las restricciones que definen a un sistema RESTful serían:

* Cliente-servidor: esta restricción mantiene al cliente y al servidor débilmente acoplados. Esto quiere decir que el cliente no necesita conocer los detalles de implementación del servidor y el servidor se “despreocupa” de cómo son usados los datos que envía al cliente.
* Sin estado: aquí decimos que cada petición que recibe el servidor debería ser independiente, es decir, no es necesario mantener sesiones.
* Cacheable: debe admitir un sistema de almacenamiento en caché. La infraestructura de red debe soportar una caché de varios niveles. Este almacenamiento evitará repetir varias conexiones entre el servidor y el cliente para recuperar un mismo recurso.
* Interfaz uniforme: define una interfaz genérica para administrar cada interacción que se produzca entre el cliente y el servidor de manera uniforme, lo cual simplifica y separa la arquitectura. Esta restricción indica que cada recurso del servicio REST debe tener una única dirección, “URI”.
* Sistema de capas: el servidor puede disponer de varias capas para su implementación. Esto ayuda a mejorar la escalabilidad, el rendimiento y la seguridad.

# 12. Ingeniería inversa y re-ingeniería

## 12.1. Ingeniería inversa

La ingeniería inversa es el proceso llevado a cabo con el objetivo de obtener información o un diseño a partir de un producto, con el fin de determinar cuáles son sus componentes y de qué manera interactúan entre sí y cuál fue el proceso de fabricación.

El método se denomina así porque avanza en dirección opuesta a las tareas habituales de ingeniería, que consisten en utilizar datos técnicos para elaborar un producto determinado. En general, si el producto u otro material que fue sometido a la ingeniería inversa fue obtenido en forma apropiada, entonces el proceso es legítimo y legal. De la misma forma, pueden fabricarse y distribuirse, legalmente, los productos genéricos creados a partir de la información obtenida de la ingeniería inversa.

La ingeniería inversa es un método de resolución. Aplicar ingeniería inversa a algo supone profundizar en el estudio de su funcionamiento, hasta el punto de que podamos llegar a entender, modificar y mejorar dicho modo de funcionamiento.

La aplicación de ingeniería inversa nunca cambia la funcionalidad del producto objeto de la aplicación, sino que permite obtener productos que indican cómo se ha construido el mismo. Su realización permite obtener los siguientes beneficios:

* Reducir la complejidad del sistema: al intentar comprender el sistema se facilita su mantenimiento y la complejidad existente disminuye.
* Generar diferentes alternativas: del punto de partida del proceso, principalmente código fuente, se generan representaciones gráficas lo que facilita su comprensión.
* Recuperar y/o actualizar la información.
* Detectar efectos laterales: los cambios que se puedan realizar en un sistema puede conducirnos a que surjan efectos no deseados, esta serie de anomalías puede ser detectados por la ingeniería inversa.
* Facilitar la reutilización: por medio de la ingeniería inversa se pueden detectar componentes de posible reutilización de sistemas existentes, pudiendo aumentar la productividad, reducir los costes y los riesgos de mantenimiento.

## 12.2. Re-ingeniería

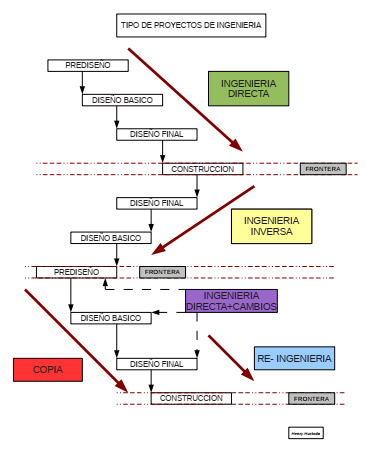
Reingeniería en un concepto simple es el rediseño de un proceso en un negocio o un cambio drástico de un proceso. A pesar que este concepto resume la idea principal de la reingeniería esta frase no envuelve todo lo que implica la reingeniería. Reingeniería es comenzar de cero, es un cambio de todo o nada, además ordena la empresa alrededor de los procesos. La reingeniería requiere que los procesos fundamentales de los negocios sean observados desde una perspectiva transfuncional y en base a la satisfacción del cliente.

Una definición rápida de reingeniería es "comenzar de nuevo". Reingeniería también significa el abandono de viejos procedimientos y la búsqueda de trabajo que agregue valor hacia el consumidor.

La definición más aceptada actualmente es la siguiente "La Reingeniería es el replanteamiento fundamental y el rediseño radical de los procesos del negocio para lograr mejoras dramáticas dentro de medidas críticas y contemporáneas de desempeño, tales como costo, calidad, servicio y rapidez". (Hammer 1994)

La implementación paulatina de un enfoque basado en procesos permite a una organización:

* Establecer indicadores de gestión para los procesos básicos de la organización e indicadores de resultados Simplificar y estandarizar los flujos de operación.
* Eliminar actividades sin valor agregado
* Mejorar los flujos de información
* Reducir tiempos de operación
* Mantener los procesos focalizados en el cliente
* Mejorar la calidad del servicio



# 13. Descripción de herramientas de desarrollo

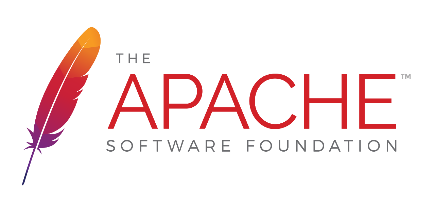
## 13.1. PhpStorm



PhpStorm de JetBrains es un PHP IDE inteligente y profesional que proporciona a los desarrolladores una mezcla de potentes herramientas inteligentes, hábitos útiles y mejores prácticas para el desarrollo de PHP, todos orientados hacia sacar provecho a la productividad del desarrollador.

* Asistencia de codificación inteligente
* Análisis de Calidad del código
* Entorno de desarrollo
* HTML/CSS/JavaScript Editor
* Fácil depuración y pruebas
* Refactorización rápida y segura

## 13.2. Apache



El servidor HTTP Apache es un servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual según la normativa RFC 2616. Cuando comenzó su desarrollo en 1995 se basó inicialmente en código del popular NCSA HTTPd 1.3, pero más tarde fue reescrito por completo. Su nombre se debe a que alguien quería que tuviese la connotación de algo que es firme y enérgico, pero no agresivo, y la tribu Apache fue la última en rendirse al que pronto se convertiría en gobierno de Estados Unidos, y en esos momentos la preocupación de su grupo era que llegasen las empresas y "civilizasen" el paisaje que habían creado los primeros ingenieros de internet. Además, Apache consistía solamente en un conjunto de parches a aplicar al servidor de NCSA. En inglés, a patchy server (un servidor "parcheado") suena igual que Apache Server.

* Modular
* Código abierto
* Multi-plataforma
* Extensible
* Popular (fácil conseguir ayuda/soporte)

## 14.3. Enterprise Architect



Sparx Systems Enterprise Architect es una herramienta de modelado y diseño visual basada en OMG UML. La plataforma admite: el diseño y la construcción de sistemas de software; modelar procesos comerciales; y modelado de dominios basados ​​en la industria. Las empresas y organizaciones lo usan no solo para modelar la arquitectura de sus sistemas, sino también para procesar la implementación de estos modelos en todo el ciclo de vida de desarrollo de aplicaciones.

* Modelado y análisis de negocios y datos
* Gestión de pruebas y simulación
* Análisis de ejecución visual
* Gestión del proyecto
* Integración con otras herramientas

## 13.4. Git



Git es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente. Su propósito es llevar registro de los cambios en archivos de computadora y coordinar el trabajo que varias personas realizan sobre archivos compartidos.

* Compartición selectiva
* Velocidad y seguridad
* Ramificación
* Convergencia

## 13.5. Visual Studio Code



Visual Studio Code es un editor de código fuente desarrollado por Microsoft para Windows , Linux y macOS. Incluye soporte para la depuración, control integrado de Git , resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código. También es personalizable, por lo que los usuarios pueden cambiar el tema del editor , los atajos de teclado y las preferencias. Es gratuito y de código abierto, aunque la descarga oficial está bajo software propietario.

* IntelliSense, una función que nos permite desde resaltar la sintaxis de nuestros proyectos.
* Cuenta con un avanzado depurador de código.
* Se integra en Git. Nos permite versionar nuestros trabajos fácilmente desde esta plataforma.
* Gracias a sus módulos, es compatible con prácticamente cualquier lenguaje de programación.

## 13.6. Postman



Postman surgió originariamente como una extensión para el navegador Google Chrome. A día de hoy dispone de aplicaciones nativas para MAC y Windows y están trabajando en una aplicación nativa para Linux (disponible en versión beta).

Está compuesto por diferentes herramientas y utilidades gratuitas (en la versión free) que permiten realizar tareas diferentes dentro del mundo API REST: creación de peticiones a APIs internas o de terceros, elaboración de tests para validar el comportamiento de APIs, posibilidad de crear entornos de trabajo diferentes (con variables globales y locales), y todo ello con la posibilidad de ser compartido con otros compañeros del equipo de manera gratuita (exportación de toda esta información mediante URL en formato JSON).

# 14. Proceso del software

## 14.1. Modelo de requisito

## 14.2. Modelo de arquitectura del software a nivel del código

## 14.3. Modelo de arquitectura del software a nivel de despliegue

## 14.4. Modelo de persistencia en el servidor y móvil

# 16. Referencias