

**Universidad Tecnológica del Centro de Veracruz**

**Programa educativo de Tecnologías de la información y la comunicación**

**Roca 2**

**Presenta:**

* **Contreras Ortiz José Manuel**
* **Elotlán Hernández Ana Karina**
* **Hernández Chávez Celia**
* **Sánchez Gómez Estuardo**

**Cuitláhuac, Veracruz a 17 de junio del 2015**

**Versiones para “Descripción del proyecto”**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Descripción** | **Autor** |
| 16/06/15 | 4.0 | Se compila la información de la empresa y proyecto en un solo documento. | José Manuel Contreras. |

ÍNDICE

[Antecedentes de la empresa 4](#_Toc423013988)

[Descripción de la problemática 5](#_Toc423013989)

[Descripción del producto y entregables. 6](#_Toc423013990)

[Objetivos 7](#_Toc423013991)

[Objetivo general 7](#_Toc423013992)

[Objetivos específicos 7](#_Toc423013993)

[Justificación 8](#_Toc423013994)

[Alcance 9](#_Toc423013995)

[Limitaciones 9](#_Toc423013996)

[Herramientas 11](#_Toc423013997)

[PostgreSQL 11](#_Toc423013998)

[Componentes más importantes en un sistema PostgreSQL 11](#_Toc423013999)

[Características 12](#_Toc423014000)

[Características Generales 12](#_Toc423014001)

[Desarrollo 13](#_Toc423014002)

[SQL 13](#_Toc423014003)

[JAVA 14](#_Toc423014004)

[Java Virtual Machine (JVM) 15](#_Toc423014005)

[¿Qué se necesita para programar en Java? 15](#_Toc423014006)

[Scene Builder 17](#_Toc423014007)

[Características del hardware 18](#_Toc423014008)

[Marco Teórico 19](#_Toc423014009)

[Bases de datos 19](#_Toc423014010)

[Tipos de relaciones (Cardinalidad) en las relaciones 22](#_Toc423014011)

[Normalización 23](#_Toc423014012)

[Java 24](#_Toc423014013)

[Java como lenguaje de programación 24](#_Toc423014014)

[Paradigma de programación orientada a objetos 25](#_Toc423014015)

[Metodología de desarrollo Moprosoft 26](#_Toc423014016)

[Introducción 26](#_Toc423014017)

[Ventajas 28](#_Toc423014018)

[Desventajas 28](#_Toc423014019)

[Fases 28](#_Toc423014020)

[Diagrama de la metodología 31](#_Toc423014021)

[Bibliografía 32](#_Toc423014022)

# Antecedentes de la empresa

La Roca 2 es una micro empresa fundada en el año 2012 por Gerardo Sánchez Estrada actual propietario del negocio quien se ha dedicado a la venta y reparación de motocicletas así como de la venta de refacciones.

Durante el primer año resultó muy problemático para el dueño tener clientes ya que en el sitio donde se estableció antes de él también era taller mecánico el cual dio como resultado que el nuevo negocio que se pretendía establecer fuera adquiriendo mala reputación. Fue con el tiempo que esta empresa desempeñó muy bien su trabajo rompiendo así con los prejuicios iniciales que se gestaron en un comienzo, de este modo cada vez iba más clientela no solamente de la zona local sino también de otras regiones, incluyendo empresas reconocidas de la región.

Pese a ello hubo que enfrentar ciertas complicaciones como el hecho de no contar con un registro hacendario o el no utilizar ninguna herramienta de tipo tecnológica para agilizar sus operaciones, toda la administración era llevada por el fundador y propietario debido a las consecuencias que pudiera tener el relevar esas responsabilidades.

# Descripción de la problemática

La Roca 2 es una micro empresa que está dándose a conocer y gracias a esto es que ha ganado mercado, una desventaja que tiene esta empresa, es su control de ventas, ya que aunque cuenta con una aplicación que le facilita llevar a cabo sus operaciones diarias no tiene implementada la opción de reportes.

Y no ha recibido el mantenimiento adecuado, para seguir funcionando de manera adecuada, además de que no cuenta con un respaldo de base de datos, lo que a largo plazo le podría ocasionar un grave problema.

Por lo tanto se pretende mejorar dicha aplicación implementando reportes con los cuales le sea más fácil al cliente y al dueño ver la información que solicite y generar tickes o facturas si así lo desea, de esta manera ambos saldrán beneficiados.

También se pretende mejorar en la totalidad la interfaz de la aplicación de manera que sea para el usuario más fácil de usar y de realizar búsquedas de información en la base de datos, ya que se implementaran filtros para poder realizar consultas más rápidamente y ofrecer al dueño los resultados que espera.

# Descripción del producto y entregables.

|  |  |  |  |
| --- | --- | --- | --- |
| Descripción de los entregables para el cliente | | | |
| Entregable | Descripción | Fecha | Lugar |
| Garantía | Se establecen los términos bajos los cuales ambas partes se responsabilizan del uso e implementación del proyecto. | 27/08/2015 | Fortín, Ver. |
| Base de datos | Define el modelo lógico y físico bajo el cual deberá operar la aplicación a desarrollar. | 28/08/2015 | Fortín, Ver. |
| Minutas de reunión | Reportes que contienen los detalles relativos a las reuniones realizas por el equipo de trabajo responsable del proyecto. | 27/08/2015 | Fortín, Ver. |
| Aplicación final | Contiene una versión definitiva de la aplicación ofreciendo las funcionalidades establecidas en su momento. | 28/08/2015 | Fortín, Ver. |
| Manual de usuario | Es el documento destinado a ser el manual que contiene las especificaciones de uso de la aplicación por parte de los usuarios comunes. | 28/08/2015 | Fortín, Ver. |
| Manual técnico | Es el documento que detalla de manera más amplia y técnica el funcionamiento de la aplicación para su posterior mantenimiento. | 28/08/2015 |  |

# Objetivos

## Objetivo general

Mejorar y optimizar una aplicación de escritorio programada en JavaFx para entornos MS Windows para la administración y automatización de los procesos administrativos de la empresa dedicada a la reparación de motor la “Roca 2” en lo respectivo al manejo de inventario, registro de empleados y servicios ofrecidos, que cumpla los estándares mínimos de seguridad de acceso a los datos y sea fácil de usar para el usuario convencional.

## Objetivos específicos

* Validar los requisitos levantados.
* Depurar errores presentes en la aplicación.
* Crear un repositorio en GitHub y organizarlo para almacenar proyecto y documentación.
* Mejorar el diseño de base de datos y portarla a PostgresSQL.
* Implementar restricciones en la base de datos no existentes en las primeras versiones.
* Cambiar el sistema de autentificación implementando conexión a base de datos.
* Sustituir controles genéricos por controles especializados (Ej. TextField por PasswordField).
* Revisar que el diseño se ajuste correctamente a MVC
* Implementar un sistema de encriptación para datos sensibles.
* Sustituir consultas directas en el código Java por funciones SQL.

# Justificación

En la actualidad el taller la “Roca 2” trabaja con una aplicación de escritorio donde lleva a cabo todas las operaciones de la empresa, pero necesita que se implemente la opción de generación de reportes para que le sea más fácil al dueño realizar consultas y tenerlas listas en cualquier momento.

Los reportes son cada vez más necesarios, puesto que la demanda de gestión automática (prácticamente en todos los ámbitos) y las posibilidades de los productos que ofrecen, va creciendo día a día, lo que conlleva a que se haga imprescindible una aplicación que permita la generación de reportes para la empresa que lo solicita.

Con esta aplicación se pretende evitarse problemas futuros, ya que tanto el cliente como el dueño de la empresa tendrán como demostrar si se realizó alguna operación inadecuada o no, por ejemplo, que a un cliente se le haya cobrado dinero de más, el con su ticked puede demostrar cuantos productos fueron los que en realidad compro y cuál fue el servicio por el que le están cobrando, incluso saber si le están cobrando lo debido.

Además de que se pretende darle el mantenimiento adecuado a la aplicación para que siga teniendo un correcto funcionamiento y le siga siendo de gran ayuda a los usuarios de la aplicación.

# Alcance

Una de las principales metas para mejorar la aplicación es implementar medidas se seguridad de acceso en base a datos dinámicos, es decir, cuya información para su utilización esté almacenada fuera de la estructura interna de la misma permitiendo su modificación por parte del personal autorizado.

Mejorar el control de los productos en el inventario, empleados y clientes, entre otros, permitir el borrado lógico de los registros por cuestiones de seguridad de la información, por lo mismo se deberá implementar un sistema de cifrado o encriptación que permita la protección de la información manipulada.

Además al contar con un sistema de gestión de usuario con conexión a base de datos el propietario quien se encargaba del todo la administración podrá cambiar privilegios de sus empleados y permitir el acceso a nuevos usuarios.

# Limitaciones

Originalmente la aplicación estaba diseñada para ser ejecutada en entornos de escritorio Windows, Linux y Mac, sin contar con ninguna versión a implementar vía web, debido a las cuestiones del lenguaje de programación utilizado y a la estructura de directorios distintas, además del lapso disponible para el desarrollo del producto, la aplicación solo podrá ser ejecutada eficazmente en plataformas MS Windows.

Se hará un rediseño para que esta funcione de manera local, una vez instalado solo podrá ser manipulado accediendo al equipo donde este se encuentre, no podrá ser accesible ni siquiera dentro de una red de área local, aunque en determinadas circunstancias será posible acceder a los binarios del programa su funcionamiento estará ligado al servicio de base de datos que se esté ejecutando.

Debido al lenguaje de programación a utilizar la aplicación requerirá la previa instalación de Java JRE versión 8, por tanto no podrá ser ejecutada bajo ningún sistema operativo que no se encuentre previamente preparado para ello.

Una de las prioridades en el desarrollo de este proyecto es modificar las interfaces para que sean más sencillas e intuitivas, de esta manera se facilitará el uso a los usuarios comunes, sin embargo esto implica que se sacrificará el acceso a funciones avanzadas para evitar el riesgo de que un usuario, en su desconocimiento, corrompa su buen funcionamiento.

El sistema de autentificación por medio de un usuario y contraseña será la única manera de acceder a las funciones del programa, se removerán funcionalidades adicionales que permitían otros medios de acceso, no se contará con ningún otro medio como podría ser el uso de cuentas en redes sociales, cuentas de correo o verificación telefónica.

# Herramientas

## PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado y en sus últimas versiones no tiene nada que envidiarle a otras bases de datos comerciales.

PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando.

### Componentes más importantes en un sistema PostgreSQL

* **Aplicación cliente**: Esta es la aplicación cliente que utiliza PostgreSQL como administrador de bases de datos. La conexión puede ocurrir vía TCP/IP o sockets locales.
* **Demonio postmaster**: Este es el proceso principal de PostgreSQL. Es el encargado de escuchar por un puerto/socket por conexiones entrantes de clientes. También es el encargado de crear los procesos hijos que se encargaran de autentificar estas peticiones, gestionar las consultas y mandar los resultados a las aplicaciones clientes.
* **Ficheros de configuración**: Los 3 ficheros principales de configuración utilizados por PostgreSQL, postgresql.conf, pg\_hba.conf y pg\_ident.conf
* **Procesos hijos postgres**: Procesos hijos que se encargan de autentificar a los clientes, de gestionar las consultas y mandar los resultados a las aplicaciones clientes.
* **PostgreSQL share buffer cache**: Memoria compartida usada por PostgreSQL para almacenar datos en caché.
* **Write-Ahead Log (WAL)**: Componente del sistema encargado de asegurar la integridad de los datos (recuperación de tipo REDO)
* **Kernel disk buffer cache**: Caché de disco del sistema operativo
* **Disco**: Disco físico donde se almacenan los datos y toda la información necesaria para que PostgreSQL funcione.

### Características

La última serie de producción es la 9.3. Sus características técnicas la hacen una de las bases de datos más potentes y robustos del mercado. Su desarrollo comenzó hace más de 16 años y durante este tiempo, estabilidad, potencia, robustez, facilidad de administración e implementación de estándares han sido las características que más se han tenido en cuenta durante su desarrollo. PostgreSQL funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema.

### Características Generales

* Es una base de datos 100%
* Integridad referencial
* Tablespaces
* Nested transactions (savepoints)
* Replicación asincrónica/sincrónica / Streaming replication - Hot Standby
* Two-phase commit
* PITR - point in time recovery
* Copias de seguridad en caliente (Online/hot backups)
* Unicode
* Juegos de caracteres internacionales
* Regionalización por columna
* Multi-Version Concurrency Control (MVCC)
* Multiples métodos de autentificación
* Acceso encriptado via SSL
* Actualización in-situ integrada (pg\_upgrade)
* SE-postgres
* Completa documentación
* Licencia BSD
* Disponible para Linux y UNIX en todas sus variantes (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64) y Windows 32/64bit.

### Desarrollo

* Funciones/procedimientos almacenados (stored procedures) en numerosos lenguajes de programación, entre otros PL/pgSQL (similar al PL/SQL de Oracle), PL/Perl, PL/Python y PL/Tcl
* Bloques anónimos de código de procedimientos (sentencias DO)
* Numerosos tipos de datos y posibilidad de definir nuevos tipos. Además de los tipos estándares en cualquier base de datos, tenemos disponibles, entre otros, tipos geométricos, de direcciones de red, de cadenas binarias, UUID, XML, matrices, etcétera.
* Soporta el almacenamiento de objetos binarios grandes (gráficos, videos, sonido).
* APIs para programar en C/C++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, PHP, Lisp, Scheme, Qt y muchos otros.

### SQL

* SQL92,SQL99,SQL2003,SQL2008
* Llaves primarias (primary keys) y foráneas (foreign keys)
* Check, Unique y Not null constraints
* Restricciones de unicidad postergables (deferrable constraints)
* Columnas auto-incrementales
* Indices compuestos, únicos, parciales y funcionales en cualquiera de los metodos de almacenamiento disponibles, B-tree, R-tree, hash ó GiST
* Sub-selects
* Consultas recursivas
* Funciones 'Windows'
* Joins
* Vistas (views)
* Disparadores (triggers) comunes, por columna, condicionales.
* Reglas (Rules)
* Herencia de tablas (Inheritance)
* Eventos LISTEN/NOTIFY

## JAVA

Java es uno de los lenguajes más difíciles de aprender, quizás no tanto por su sintaxis o por el hecho de que está orientado a objetos, sino más bien por su amplitud. Dentro de Java hay una biblioteca de clases (librerías) para hacer cualquier tipo de cosa. Es muy difícil que se desarrolle cualquier tipo de aplicación y comience desde cero en Java, al contrario, siempre se apoyará en algo ya escrito, que implementa las bases de aquello que quiere construir.

Java es muy grande y aunque represente una dificultad en el aprendizaje, es también una de sus ventajas. De hecho, se podría decir que, aunque un desarrollador tenga muchos años de experiencia programando, es prácticamente imposible que conozca todas las áreas para las que Java se puede utilizar, ya que son muchas y muy diferentes. Normalmente, las clases (librerías) que se utilizan cuando se realiza un tipo de programa son diferentes a las que se utilizan para otro programa y lo habitual es que los proyectos solo requieran trabajar con un pequeño conjunto de los recursos que existen en Java.

### Java Virtual Machine (JVM)

Java es un lenguaje multiplataforma, que se ejecuta en cualquier máquina. Esto es gracias a la JVM (Java Virtual Machine) que nos permite ejecutar el código de Java en cualquier lugar para el que se haya creado dicha máquina virtual. JVM es el secreto (no tan secreto porque todo el mundo lo sabe) y la clave de Java como lenguaje multiplataforma.

La JVM hace de puente entre el código compilado de Java y la máquina donde se pretende ejecutar. Podemos verlo como un traductor, que se encarga de interpretar el código de Java, de manera que sea correctamente ejecutado en el sistema objetivo. Es decir, con Java compilamos el código y el resultado de la compilación se ejecuta en la JVM que tenga el sistema operativo donde quieres poner en marcha el programa.

Casi en cualquier hardware se puede montar un JVM y gracias a ello ser compatible con Java. De hecho, los primeros teléfonos móviles, antes de la llegada de los Smartphone, tenían la posibilidad de ejecutar juegos en Java y eso es porque simplemente se había creado una JVM para ellos.

### ¿Qué se necesita para programar en Java?

1) Java JDK

Lo primero que se necesita para poder desarrollar en Java es el "Java Development Kit". Es un software gratuito que contiene todo aquello que requiere tu máquina para trabajar con el lenguaje, tanto la JVM como las librerías para realizar programas de Java, desde los más básicos hasta los más complejos y específicos.

Para obtener el JDK necesitas descargarlo desde la página de Oracle.

Hay diferentes "entregas", "sabores" o "paquetes", de Java. Cada una tiene un nombre y a veces resulta un poco lioso por tener tantas siglas que aprenderse.

Cuando se descarga el JDK, en realidad se está descargando Java por completo, incluyendo todas las librerías y también por supuesto el Java VM (JVM). Si te descargas el JRE, en realidad estás trayéndote un subconjunto de Java. Así pues, simplemente tienes que fijarte qué es lo que estás descargando para traerte aquello que necesites.

2) JVM

Si tienes o no Java realmente no debemos preocuparnos, la JMV se instala también cuando se instala el JDK para Java.

3) IDE para desarrollo

El tercer paso sería contar con un IDE para el desarrollo. IDE son las siglas de "Integrated Development Environment" o en español, Entorno de Desarrollo Integrado.

Alternativas de IDE para Java

Sobre IDE tenemos varias opciones para Java, entre las más conocidas y usadas tenemos Eclipse, Netbeans, Jdeveloper, IntelliJ IDEA. Cualquiera de ellos puedes usarlo perfectamente, así como cualquier otro que encuentres y te sientas cómodo.

El IDE Eclipse es con diferencia el más usado (tiene en torno al 60% o 70% de tasa de uso entre los desarrolladores de Java) porque es el primero que se popularizó, gracias a que fue el que primero introdujo las extensiones por medio de plugins. Java es atractivo también porque es abierto y gratuito para cualquier tipo de uso.

Eclipse para programar en Java

Eclipse es multiplataforma, por lo que también tendrás que asegurarte de estar descargando exactamente la versión para tu sistema operativo, Linux, Mac o si estás en Windows, la versión de 32 Bit o 64 Bit que corresponda con el tipo de procesador de tu PC.

Eclipse tiene una particularidad y es que no requiere instalación, al contrario de lo que estamos acostumbrados en muchos sistemas. Realmente lo que descargas es el programa entero, ya "paquetizado" para que puedas usarlo. Simplemente tienes que descomprimir el contenido de tu fichero y dejarlo en cualquier localización de tu disco duro, el escritorio, la raíz del disco, un disco externo o donde desees. Una vez descomprimido, simplemente tendrás que hacer un doble clic en el ejecutable (eclipse.exe) para poner en marca el IDE.

Como puedes ver, Eclipse se instala como lo que podríamos decir una versión "portable" del software, porque todo lo que necesita el programa para funcionar está en la carpeta que has descomprimido. Si te llevas la carpeta a cualquier otra localización, seguirá funcionando de la misma manera. Incluso si tienes que cambiar de ordenador, simplemente te llevas la carpeta a la nueva máquina y funcionará perfectamente, junto con todos tus proyectos.

Otra de las ventajas que nos proporciona el estilo de instalación de Eclipse, sin el típico asistente, es que puedes tener varias distribuciones de Eclipse, configuradas de distinta manera, para varios tipos de lenguajes o varios proyectos, sin que se peguen la una con la otra.

## Scene Builder

Hay dos formas de crear la interfaz de usuario. Programándolo en Java o mediante un archivo XML. Aunque existe información relativa a ambos métodos. Si usamos XML (archivo con la extensión .fxml) puede que se encuentre más claro mantener el controlador y la vista separados entre sí.

Además, se puede usar la herramienta de edición visual Scene Builder, la cual nos evita tener que trabajar directamente con el XML.

Una nueva herramienta para diseñar y crear contenidos en JavaFX, que básicamente es un editor de archivos FXML.

Es el comienzo de una herramienta más completa de tipo RAD (desarrollo rápido de aplicaciones) para JavaFX, con capacidades de construir un GUI mediante arrastrar y soltar y eventualmente conexión de datos mediante el diseño en Scene Builder.

### Características del hardware

La aplicación se podrá ejecutar en algún equipo que cuente con el sistema operativo Windows 7 u Windows 8 y el equipo debe contar mínimo con:

* RAM: 128 MB
* Espacio en disco: 124 MB para JRE; 2 MB para Java Update
* Procesador: Mínimo Pentium 2 a 266 MHz
* Exploradores: Firefox, Chrome

Estas especificaciones del equipo de cómputo servirán para el buen funcionamiento de la aplicación y para mantener la eficiencia del software que se pretende instalar.

# Marco Teórico

## Bases de datos

A través de los años los sistemas de administración de Bases de Datos han evolucionado hacia Sistemas de Administración de Base de Datos Relacionales (RDBMS). Una base de datos relacional es un modelo organizado de entidades que posee características que tienen relaciones entre ellas. Una base de datos relacional bien diseñada provee información de un negocio o un proceso y su uso más común es para almacenar y recuperar información. Entre las mayores ventajas de RDBMS están la forma en la que almacena y recupera información y cómo mantiene la integridad de la misma. Las estructuras RDBMS son fáciles de comprender y construir, pues son lógicamente representadas utilizando Diagramas Entidad-Relación. Las bases de datos relacionales tienen las siguientes características principales:

• Estructuras. Son objetos que almacenan o acceden a los datos de la base de datos (Tablas, vistas e índices).

• Tabla. Es un objeto que almacena datos en forma de filas y columnas. Cada tabla tiene una o más columnas y filas. Las columnas guardan una parte de la información sobre cada elemento que queremos guardar en la tabla, cada fila de la tabla conforma un registro.

Los datos de una tabla contienen valores atómicos, es decir que contiene elementos indivisibles.

• Integridad. La integridad de la base de datos se refiere a la validez y la consistencia de los datos.

• Acceso concurrente. Los sistemas manejadores de base de datos (RDBMS) controlan el acceso concurrente a una base de datos por parte de múltiples usuarios.

• Flexibilidad. La base de datos ofrece fácilmente distintas vista en función de los usuarios y aplicación.

• Independencia física. Permite modificar el esquema físico sin tener que rescribir los programas de la aplicación. Las modificaciones a nivel físico son necesarias ocasionalmente para mejorar el funcionamiento del sistema.

• Independencia lógica. Permite modificar el esquema conceptual sin que se tenga que alterar el programa de aplicación. Estos cambios a nivel conceptual son necesarios cuando la estructura lógica de la base de datos es alterada. Esta independencia es más difícil de implementar, ya que los programas de aplicación dependen de la estructura lógica de los datos que accedan.

• Facilidad de uso. Los usuarios tendrán fácil acceso a los datos. Las complejidades internas son ajenas al usuario, gracias al sistema de administración de la base.

• Redundancia controlada. Los datos serán almacenados una sola vez excepto cuando existan razones técnicas o económicas que aconsejen el almacenamiento redundante.

• Seguridad de acceso. Se evitará el acceso no autorizado de datos. Los mismos podrán estar sujetos a diferentes restricciones de acceso para distintos usuarios.

• Operaciones. Son acciones usadas para definir las Estructuras o manipular los datos de las mismas (SELECT, CREATE)

• Reglas de integridad. Gobiernan los tipos de acciones permitidas en los datos y la estructura de la Base de Datos (BD). Protegen los datos y estructuras de la BD. (Llaves primarias y foráneas).

• Identificador único. No pueden existir dos tablas con el mismo nombre, así como no pueden existir dos columnas con el mismo nombre en una misma tabla y los valores almacenados en una columna deben ser del mismo tipo de dato.

• Clave única. Cada tabla puede tener uno o más campos cuyos valores identifican de forma única cada registro de dicha tabla, es decir, no pueden existir dos o más registros diferentes cuyos valores en dichos campos sean idénticos. Este conjunto de campos se llama clave única.

• Clave primaria. Una clave primaria es una clave única elegida entre todas las candidatas que define unívocamente a todos los demás atributos de la tabla, para especificar los datos que serán relacionados con las demás tablas. La forma de hacer esto es por medio de claves foráneas. Sólo puede existir una clave primaria por tabla y ningún campo de dicha clave puede contener valores NULL.

• Dominios. Un dominio describe un conjunto de posibles valores para cierto atributo. Como un dominio restringe los valores del atributo, puede ser considerado como una restricción. Matemáticamente, atribuir un dominio a un atributo significa "todos los valores de este atributo deben de ser elementos del conjunto especificado".

• Normalización. Las bases de datos relacionales pasan por un proceso al que se le conoce como normalización, el resultado de dicho proceso es un esquema que permite que la base de datos sea usada de manera óptima.

Introducción a las metodologías de diseño

Un modelo es una representación de un sistema que pretende simplificar su comprensión poniendo en evidencia ciertos aspectos del sistema mientras otros son ocultados. Los modelos se utilizan para facilitar la tarea de diseño de los SI complejos, ya que facilitan pensar en lo que se está haciendo y permiten comprobar la corrección y adecuación al problema de los resultados.

Los modelos pueden tener distintos niveles de abstracción.

• El modelo físico, que describe completamente el sistema: circulación y tratamiento de la información, elementos informáticos y elementos manuales. Para la BD el modelo físico representa la organización de la información sobre los soportes de almacenamiento.

• El modelo lógico, que describe las informaciones y las manipulaciones a que son sometidas. Este modelo hace abstracción de los soportes materiales de almacenamiento. El modelo lógico sobre una BD representa la definición de la información sobre el SGBD elegido para el desarrollo del SI.

• El modelo conceptual, que describe el contenido subyacente al modelo lógico, esto es, el significado de las informaciones y las relaciones que las unen. Este modelo hace abstracción de las manipulaciones de la información.

### Tipos de relaciones (Cardinalidad) en las relaciones

Una de las formas para establecer una restricción en las relaciones es a través de la cardinalidad, este concepto nos permite establecer el número máximo de instancias en un conjunto de entidades que está relacionado con una única instancia en el otro conjunto de entidades.

• Uno a uno. Cada elemento de una entidad es representada con un elemento único en otra entidad

• Uno a muchos. Cada elemento de una entidad puede ser representada por muchos elementos en otra entidad

• Muchos a muchos. Un elemento de una entidad es representado por uno o más elementos en otra entidad y cada elemento de la segunda entidad puede ser representada por uno o más elementos de la primera entidad

### Normalización

Normalización es un proceso que clasifica relaciones, objetos, formas de relación y demás elementos en grupos, en base a las características que cada uno posee. Si se identifican ciertas reglas, se aplica una categoría; si se definen otras reglas, se aplicará otra categoría.

Cuando se han establecido las relaciones entre entidades, se debe realizar la normalización del diseño, esto es, eliminar la información redundante de las entidades. Las reglas principales para normalización son las siguientes:

• Primera Forma Normal (1FN). Una tabla está en Primera Forma Normal sólo si:

• Todos sus atributos son atómicos. Un atributo es atómico si los elementos del dominio son indivisibles, mínimos.

• Una tabla contiene una llave primaria.

• La tabla no contiene atributos nulos.

• Si no posee ciclos repetitivos. Una columna no puede tener múltiples valores. Los datos son atómicos. (Si a cada valor de X le pertenece un valor de Y, entonces a cada valor de Y le pertenece un valor de X).

• Segunda Forma Normal (2FN). Dependencia Funcional. Una relación está en 2FN si está en 1FN y si los atributos que no formen parte de ninguna clave dependen de forma completa de la clave principal, es decir que no existan dependencias parciales.

• Tercera Forma Normal (3FN). La tabla se encuentra en 3FN si esta en 2FN y cada atributo que no forma parte de ninguna clave, depende directamente y no transitivamente de la clave primaria.

## Java

### Java como lenguaje de programación

Un lenguaje de programación es un idioma artificial diseñado para expresar procesos que pueden ser llevados a cabo por máquinas como las computadoras.

Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana.

Está formado por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Al proceso por el cual se escribe, se prueba, se depura, se compila y se mantiene el código fuente de un programa informático se le llama programación.

También la palabra programación se define como el proceso de creación de un programa de computadora, mediante la aplicación de procedimientos lógicos, a través de los siguientes pasos:

El desarrollo lógico del programa para resolver un problema en particular. Escritura de la lógica del programa empleando un lenguaje de programación específico (codificación del programa). Ensamblaje o compilación del programa hasta convertirlo en lenguaje de máquina, prueba y depuración del programa, desarrollo de la documentación.

Java es un lenguaje de programación y la primera plataforma informática creada por Sun Microsystems en 1995. Es la tecnología subyacente que permite el uso de programas punteros, como herramientas, juegos y aplicaciones de negocios. Java se ejecuta en más de 850 millones de ordenadores personales de todo el mundo y en miles de millones de dispositivos, como dispositivos móviles y aparatos de televisión.

### Paradigma de programación orientada a objetos

La programación orientada a objetos representa un nivel de abstracción más alto, pues de busca representar la realidad en cada uno de los objetos y los cambios en los datos deberían afectar únicamente al objeto que pertenecen y al grupo de funciones que este contiene.

Las características que hacen de la programación orientada a objetos mejorar la calidad de los programas son:

Abstracción:

En este tipo de programación la representación de los datos es mucho más cercana a la realidad.

Modularidad:

Esta característica permite dividir un sistema, la programación orientada a objetos permite una mayor modularidad.

Extensibilidad:

Los sistemas en general necesitan modificaciones, esta propiedad reduce los cambios a dos tipos, los que son del propio objeto y repercuten en este y los que son externos a él y repercuten en el resto del sistema.

Reutilización:

Gracias a la modularidad la reutilización del código es mucho mayor, la dificultad radica en el diseño de componentes genéricos y sencillos que puedan usarse en varias partes de la aplicación.

Clases y objetos:

Los objetos se forman por datos y operaciones que pueden modificar estos datos.

La clase es como un molde para crear objetos. El objeto es la clase de la instancia.

Herencia y polimorfismo:

El polimorfismo se aplica a una sola variable que se refiere a objetos de diferentes clases. Esto significa que al crear un objeto de una clase puede referirse a cualquier objeto de las clases que desciendan de ella.

Clases y metodos abstractos:

Un método abstracto se inicializa con la palabra abstract y no tiene cuerpo. Una clase es abstracta si tiene un método de ese tipo o en caso de que no implemente un método abstracto heredado.

Una superclase abstracta no puede ser instanciada, el objetivo de ser abstracto es que sus clases implementen sus metodos abstractos.

## Metodología de desarrollo Moprosoft

### Introducción

Desde la aparición de las computadoras digitales, el desarrollo de software ha sido considerado un arte por algunos y una ciencia por otros. Con el avance en la tecnología digital el software se ha vuelto a su vez más complejo, más poderoso y por ende con más probabilidad de fracaso. Por esto, cada día es más claro que el proceso de desarrollo de software debe seguir reglas que permitan generar un producto que cumpla sus requerimientos de manera confiable y al mínimo costo. La Ingeniería de Software, definida por la IEEE como “la aplicación de un enfoque sistemático, disciplinado y cuantificable hacia el desarrollo, operación y mantenimiento del software”, presenta a los desarrolladores de software las técnicas necesarias para reducir la posibilidad de fracaso en la generación del producto software. La ingeniería de software contiene elementos que pueden agruparse en varias capas. En la capa básica se encuentran los componentes que permiten dar un enfoque de Calidad al desarrollo completo. Sobre éstos se encuentran los procesos, es decir, las guías que establecen los pasos probados para la generación de los productos que eventualmente permiten la construcción del software. Estos procesos pueden aplicarse siguiendo métodos específicos, los cuales dependen del tipo de software a desarrollarse. Para poder llevar a cabo todo esto, se utilizan herramientas que incluyen lenguajes de programación, software para controlar configuración, para controlar el proyecto etc. La teoría que apoya los componentes de cada una de estas capas es inmensa. En este artículo nos concentraremos en la capa de los procesos. Mostraremos que los procesos bien implementados llevan al éxito en el desarrollo de software y hablaremos de un modelo de proceso de desarrollo de software, conocido como Moprosoft, que ha sido diseñado para la pequeña y mediana industria mexicana. Asimismo, explicaremos brevemente a la norma mexicana que desde el 2005 implementa al modelo Moprosoft y permite a las empresas mexicanas obtener una validación oficial del nivel de capacidad de madurez con que cuentan sus procesos.

Moprosoft

Es un modelo para la mejora y evaluación de los procesos de desarrollo y mantenimiento y productos de software.

Desarrollado por la Asociación Mexicana para la Calidad en Ingeniería de Software a través de la Facultad de Ciencias de la Universidad Nacional Autónoma de México.

Moprosoft es el nombre del modelo en la comunidad universitaria y profesional y la norma técnica a la que da contenido es la NMX\_059/01-NYCE-2005 que fue declarada Norma Mexicana el 15 de agosto del 2005 con la publicación de su declaratoria en el Diario de la Federación.

El modelo destaca la importancia de la gestión de recursos, con especial relevancia en aquellos que componen el conocimiento de la organización: productos generados por proyectos, datos de los proyectos, mediciones, documentación de procesos y datos cosechados a partir del uso. Algunas de las características de Moprosoft es: Fácil de entender, práctico de aplicar en organizaciones pequeñas, específico para el desarrollo y el mantenimiento de software y aplica como norma mexicana.

### Ventajas

* Específico para el desarrollo y mantenimiento de software
* Definido como un conjunto de áreas clave de procesos
* Tiene un modelo de evaluación
* Desde 1998 empezó a popularizarse en México
* Existen organizaciones evaluadas

### Desventajas

* Es un modelo extranjero, no internacional
* No es fácil de aplicar en organizaciones pequeñas.
* Mejora no enfocada directamente a objetivos de negocio
* Evaluación es costosa y no tiene periodo de vigencia
* Se está abandonando a favor de CMM-I

### Fases

Inicio

Revisión del plan de desarrollo por los miembros del equipo de trabajo para lograr un entendimiento común del proyecto y para obtener el compromiso de su realización.

Productos generados en esta fase:

* Plan de desarrollo: cronograma, equipo de trabajo.
* Reporte de actividades: bitácora, minutas.

Requerimientos

Conjunto de actividades cuya finalidad es obtener la documentación de la especificación de requerimientos y plan de pruebas de sistema, para conseguir un entendimiento común entre el cliente y el proyecto.

Productos generados en esta fase:

* Manual de usuario (preliminar).
* Plan de pruebas del sistema.
* Especificación de requerimientos en base a la norma IEE 830
* Plan de pruebas unitarias

Análisis y diseño

Conjunto de actividades en las cuales se analizan los requerimientos especificados para producir una descripción de la estructura de los componentes de software, la cual servirá de base para la construcción.

Como resultado se obtiene la documentación de análisis y diseño y el plan de pruebas de integración.

Productos generados en esta fase:

* Plan de pruebas de integración
* Reporte de validación
* Reporte de actividades
* Diagramas: casos de uso, clases, secuencia, entidad-relación, relacional

Construcción

Conjunto de actividades para producir componentes de software que correspondan al análisis y diseño. Así como la realización de pruebas unitarias.

Como resultado se obtiene los componentes de software probados.

Productos generados en esta fase:

* Distribución de las tareas a los integrantes del equipo, según sus roles
* Modificación y construcción de software
* Reporte de actividades

Integración y pruebas

Conjunto de actividades para integrar y probar los componentes de software, basados en los planes de pruebas de integración y de sistema, con la finalidad de obtener el software que satisfaga los requerimientos especificados. Se genera la versión final del

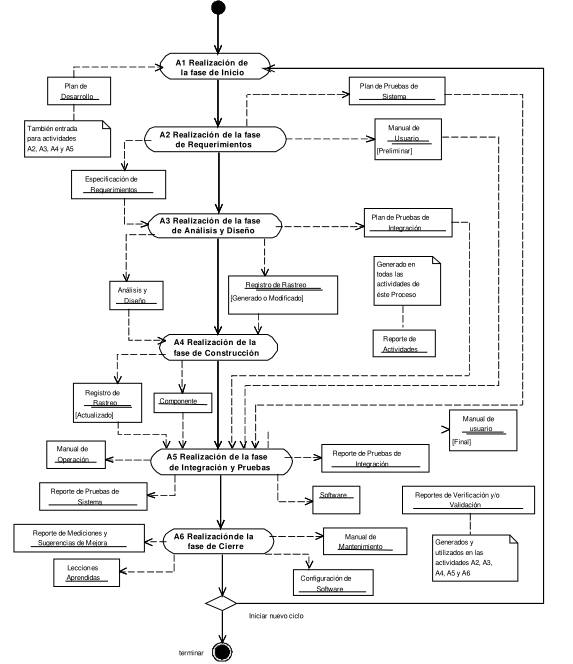
* Manual de usuario final
* Manual de operación
* Manual de mantenimiento
* Manual de instalación

Como resultado se obtiene el software probado y documentado.

Cierre

Integración final de la configuración del software generada en las fases anteriores para su entrega. Identificación y documentación de las lecciones aprendidas. Generación del reporte de mediciones y sugerencias de mejora.

### Diagrama de la metodología

****

# Bibliografía

Jacobson I., Booch G., Rumbaugh J. (2000). *El proceso unificado de desarrollo de software*. Madrid. Pearson Addison Wesley.

Ramez E., ShamKant B. N. (2007). *Fundamentos de sistemas de bases de datos. Madrid.* Pearson Addison Wesley.

Sánchez, G. C. (2007). *Fundamentos de diseño de bases de datos*. Madrid: Mc Graw Hill

Luque, R. I., Gómez N. M. A. (2002). *Bases de datos desde chen hasta codd con Oracle*. Madrid: Alfaomega Ra Ma

Kroenke, M. D. (2003). *Procesamiento de bases de datos*. México: Pearson Prentice Hall

Deitel, H. P. (2012). *Java Como programar*. México: Pearson

Eckel, B. (2007). *Piensa en java*. Madrid: Pearson Prentice Hall

Zahonero, M. I. (2010). *Programación en C, C++, Java y UML*. México: Mc Graw Hill

Deitel, J. P., Harvey, M. D. (2008). *Como programar en java.* México: Pearson Prentice Hall