**Versiones para “Marco teórico”**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Descripción** | **Autor** |
| 27/05/15 | 1.0 | El documento describe los conceptos necesarios para comprender los demás documentos. | Hernández Chávez Celia |

**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**

**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**

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**

* Basada en la norma ISO
* Facilita la comprensión del modelo
* Simplifica la relación entre el modelo de procesos y la organización
* Cuenta únicamente con 9 procesos evitando la fragmentación que se presenta en otros modelos.
* Capacidad organizacional de gestión de procesos y proyecto.

**Desventajas**

* Evaluaciones formales constantes.
* No es práctico ni fácil de usar.
* No es comprensible para los modelos ISO 9000:2000
* Mejora de procesos orientado al objetivo del negocio
* Proyectos para largos plazos

**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:

* Registro de rastreo
* 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
* Registro de rastreo

**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**

