Método para el desarrollo de una Aplicación Web a partir de un Diseño de Base de Datos con JAVA SERVER FACES (JSF)

Method for the development of a Web Application from a basic design of data with JAVA SERVER FACES (JSF)

Autores: Carrizosa Guardado Pamela, Rodríguez Echevarría Moisés, Padilla Monge Elsa Lorena, Domitsu Kono Manuel, Macías Estrada Adrián

Dirección de e-mail: mrodríguez@itson.mx

Resumen/Abstract

El desarrollo de software de aplicación para Web requiere del uso métodos actualizados y herramientas automatizadas, que ayuden a crear software de manera más ágil y que tenga las cualidades esperadas por el cliente. Partir de un Diseño de Base de Datos y el ambiente de desarrollo integrado (IDE), con la utilización del marco de trabajo (*Framework*) *Java Server Faces* (JSF) y API de Java (JPA) como herramienta para manejo de la persistencia de la misma, puede proporcionar muchas ventajas tanto al equipo de desarrollo de software como al usuario final de la aplicación.

The development of applications package for Web requires of the use up-to-date methods and automated tools, which help to create software of more agile way and than it has the qualities waited for by the client. Based on a database design and integrated development environment (IDE), and using the framework Java Server Faces (JSF) and Java API (JPA)

as a tool for managing the persistence of itself, can provide many benefits to both the software development team as the end user.

Palabras Clave: Internet, Desarrollo Web (Web Development), Base de Datos (Database), Java Server Faces

Internet es un medio de comunicación global, es una herramienta que permite la rápida obtención de información a cualquier usuario no importando su nivel social, su edad e incluso la escolaridad y la preparación del mismo. Lackerbauer (2001) afirma que actualmente Internet es algo tan habitual como la prensa, la radio, la televisión o el video. Dicho autor asegura que no hay nada que no exista en Internet y que las posibilidades que ofrece este medio son tan extensas como los intereses y las preferencias de la gente. Es por tanto un medio de comunicación que pertenece a las Tecnologías de Información (TI), las cuales ayudan a las empresas a darse a conocer en el mercado global. Hoy en día las TI son una herramienta que todas las empresas quieren y deberían tener, ya que una empresa que no implementa las TI en sus procesos es una empresa que no está a la vanguardia y que así mismo no permite que las TI realicen procesos de manera más fácil, eficiente y económica, para el beneficio la misma.

Otro punto por el cual las TI son muy importantes en las empresas es el hecho de mantenerse en un ambiente competitivo, ya que una empresa busca siempre sobresalir ante las demás para así atraer la atención de los clientes. Porter (1980), define que una empresa competitiva es aquella que anticipa los cambios en el entorno competitivo y responde a los mismos antes que sus rivales, por lo tanto las empresas que implementan las TI de la mejor manera y antes que sus rivales se encaminan hacia un grupo sobresaliente que logra ser diferenciado ante la competencia.

Una herramienta competitiva que funciona como medio de comunicación y de publicidad para los negocios son las páginas Web. Una página Web es un sitio del negocio que existe en Internet por medio del cual una organización da a conocer su estructura y los detalles

más importantes de la misma, de manera que los clientes y las demás personas puedan conocer la esencia de la empresa no importando el lugar en donde se encuentren. Para demostrar la importancia del uso de TI en las empresas, el Instituto Nacional de Estadística y Geografía (INEGI) muestra en un censo los establecimientos que usan las TI en sus procesos y relaciones con los clientes, que alrededor de 8,767 empresas en el estado de Sonora en el año 2003 usan la tecnología, este es un dato que muestra que en la actualidad la tecnología está en todas las organizaciones que quieren ser competitivas alrededor del mundo.

Ahora bien, existen dos tipos de páginas Web, las páginas Web estáticas y las páginas Web dinámicas. Las páginas Web estáticas son páginas planas en donde el negocio muestra información de la empresa, de sus productos y servicios, información administrativa, misión, visión entre otra información; se dice que son planas porque no existen procesos dentro de su funcionamiento que no vaya más allá de mostrar información e ir de una página a otra. Por otro, lado las páginas Web dinámicas son una combinación de páginas estáticas con páginas dinámicas, a esta combinación se le denomina aplicación Web; la diferencia que reside entre una aplicación Web y una página Web, es que una aplicación Web es un sitio en el que una entrada del usuario le permite interactuar e influir significativamente en el negocio, es decir puede gestionar información de negocio (De Pablos et al., 2004). Es por esto que las empresas implementan aplicaciones Web para el manejo de sus procesos, ya que por medio de las mismas los administrativos y los usuarios autorizados pueden gestionar toda la información que se maneja dentro del negocio, y a la vez existe un sitio Web en donde los clientes pueden conocer a la organización.

Para el desarrollo de las aplicaciones Web existen en el ambiente de desarrollo de software múltiples metodologías y múltiples herramientas, elegir la metodología y la herramienta correcta depende de las especificaciones y las necesidades de la empresa para quien será desarrollada. Durante el desarrollo de este ensayo se presentará un método existente para la generación de una aplicación CRUD. Las siglas CRUD significan las 4 operaciones básicas que se requieren para administrar un catálogo de objetos determinados (*Create, Read, Update, Delete*). El desarrollo partirá de un Diseño de Base de Datos y el ambiente de desarrollo integrado (IDE) que se implementa es NetBeans en su versión 6.1, con la utilización del marco de trabajo (*Framework*) *Java Server Faces* (JSF) y API de Java (JPA) como herramienta para manejo de la persistencia de la misma.

CRUD es una funcionalidad de las aplicaciones que se usa para administrar las altas y bajas. La parte de altas y bajas no es más que la administración de los activos que ingresan a la empresa y los que van de salida conforme sucede su comercialización o venta, por ejemplo los productos que son comprados y vendidos por un almacén comercial. Las funciones de una aplicación CRUD son las funciones principales requeridas por los clientes cuando se trabaja con una tabla de datos. Los datos pueden aparecer en una tabla dentro de un reporte o rellenar una lista desplegable. Independientemente de su uso los clientes frecuentemente quieren la habilidad de leer los datos en una tabla, modificar registros individuales, agregar nuevos registros y eliminar los mismos de la tabla. Esto define una típica aplicación CRUD (Myatt, 2007). Este tipo de aplicaciones funcionan básicamente haciendo movimientos en la parte de la Base de Datos (BD) o de persistencia de la aplicación.

Hay varias formas de desarrollar una aplicación CRUD de tipo Web, existen diferentes herramientas que ayudan en el desarrollo de la misma. Se encuentra por ejemplo *Visual Studio.NET*, que proporciona a los desarrolladores un ambiente de trabajo para la formulación de aplicaciones con diferentes lenguajes de programación como C#, *Visual Basic*, C++, etc. En este caso particular la herramienta que se usa para el desarrollo, como se menciona anteriormente, es *NetBeans. Netbeans* es una aplicación integrada para desarrolladores de software que trabaja en un ambiente de desarrollo de código abierto, es gratis y está conformada por todas las herramientas necesarias para el desarrollo de aplicaciones de escritorio profesionales, aplicaciones Web y aplicaciones móviles con lenguajes de programación Java, C/C++ e incluso lenguajes dinámicos como PHP, *JavaScript, Groovy y Ruby*. *NetBeans* es fácil de instalar y de usar además de que funciona sobre varias plataformas incluyendo *Windows, Linux, Mac OS X y Solaris* (NetBeans Web Site, s.f.).

El IDE de *NetBaens* permite la implementación de diferentes *Frameworks* que facilitan el buen desarrollo de las aplicaciones, en este caso JSF es un *Framework* que realiza los módulos CRUD de una forma sumamente sencilla, simplemente con el seguimiento de unos cuantos pasos que facilitan los asistentes de *NetBeans*. Esta característica hace que las aplicaciones se desarrollen de una forma fácil y es adecuada para situaciones en las que el cliente requiere la aplicación rápidamente o que simplemente requiere el desarrollo de un módulo de la aplicación que tenga que ver con altas y bajas.

Para el desarrollo de la aplicación es necesaria la ejecución de tres pasos fundamentales:

1. La creación de la BD.
2. La generación de Entidades a partir de la BD.

3. Y por último, la generación de paginas JSF en base a las entidades que participan en la aplicación.

El diseño de la BD es el primer paso que se ejecuta, dentro de este existen actividades como las siguientes:

- Crear un modelo de datos.
- Generar un diseño de BD.

Una BD es un conjunto de datos almacenados sistemática y organizadamente que proporciona a los usuarios la información que requieren en el momento preciso. Para el buen desarrollo de una BD se requiere primeramente realizar el modelado de los datos, el modelado de datos es el proceso de crear una representación lógica de la estructura de una BD (Kroenke, 2003). Existen dos modelos para el diseño de una BD, el primero es el Modelo Entidad-Relación (E-R), este modelo maneja elementos clave como las entidades, atributos, identificadores y relaciones. La figura 1 muestra un ejemplo de un diagrama E-R.

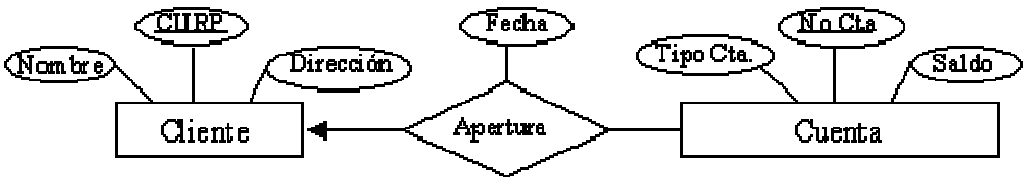

Figura 1. Representación de un Diagrama E-R.

En el modelo los rectángulos son representación de Entidades, las entidades son cosas que se pueden identificar en el ambiente de los usuarios a las cuales los usuarios requieren

darle un seguimiento, los óvalos son los atributos de la entidad y por último existe el elemento Relaciones, que son asociaciones de unas entidades con otras.

El segundo modelo de datos es el Modelo de Objeto Semántico, éste modela la percepción de los usuarios con mayor precisión que el modelo E-R. En el caso del modelo de Objeto Semántico a las entidades se les llama Objetos Semánticos. Kroenke (2003), define que los objetos semánticos son un conjunto de atributos que describen suficientemente a una identidad bien definida, a estos atributos se les llama descripción suficiente lo que significa que los atributos de los objetos son los necesarios para que los usuarios puedan trabajar correctamente. La Figura 2 muestra un ejemplo de objeto semántico:

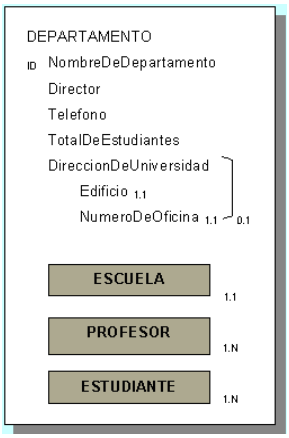

Figura 2. Representación de un objeto semántico

Al término de la definición de un modelo de la estructura de los datos se prosigue al diseño y la construcción de la BD, para esto existen diferentes métodos. Los métodos que más destacan hoy en día son el Modelo Relacional y el modelo Orientado a Objetos. El Modelo Relacional es muy conocido en lo referente a BD, ya que es el método más utilizado hoy en día para su construcción y porque es el más utilizado por los Sistemas Manejadores de Bases de Datos (SMBD) a diferencia del Modelo Orientado Objetos, que es un modelo que no es fácilmente identificado entre los SMBD por su complejidad de desarrollo. El Modelo

Relacional maneja los términos tabla, fila y columna. Las tablas están compuestas por los registros que son las filas, y también por los campos que son las columnas de la misma.

Lo más funcional para el desarrollo de la aplicación es el uso de una BD relacional por su compatibilidad con herramientas de desarrollo y SMBD. En el desarrollo de la aplicación, para la generación de la BD se cuenta con varias opciones, una es desarrollar la BD directamente en un SMBD como *MySQL* y *SQL Server*., y después crear una conexión a la misma. La otra forma es creándola desde *NetBeans*, para fines de este ensayo la BD se hará desde *NetBeans*, las actividades principales para la generación de una BD se muestran en la Figura 3.

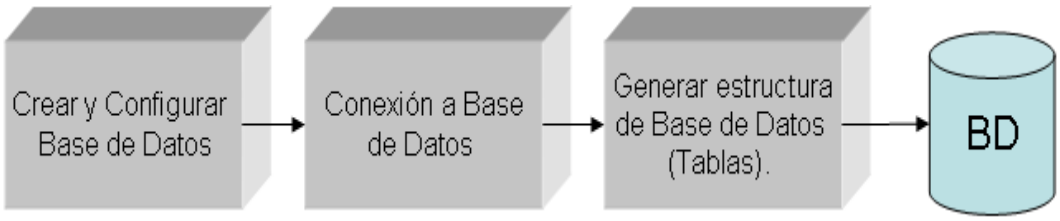

Figura 3. Procedimiento para la creación de una BD

La figura 3 muestra las actividades principales, pero a la vez dichas actividades se dividen en pequeñas unidades de acción:

1. Para la creación de la BD es requerido posicionarse en la parte de "Servicios" dentro de *NetBeans*, los servicios de *NetBeans* son un conjunto de herramientas que facilitan el acceso de las aplicaciones a servidores Web, servidores de BD, servicios Web entre otras cosas.
2. Una vez en la parte de servicios, lo que prosigue es dar "Click" derecho en el nodo *JavaBD*, y seleccionar la opción crear BD (el asistente para crear la base de datos

pide características de la misma como Nombre, Usuario, contraseña del usuario que tendrá acceso a la BD y por último la ruta en donde esta se encuentra), siguiendo estos pasos la BD ha sido creada.

3. Cuando se crea la BD un nuevo nodo aparece en la parte de *JavaBD*, para conectarse a dicho nodo, que corresponde a la BD que fue creada recientemente, es necesario dar "Click" derecho a dicho nodo y elegir la opción conectar.
4. Para la generación de las tablas se expande el nodo de la BD para ubicarse en el nodo de tablas, seguido de esto dar "Click" derecho y elegir la opción ejecutar comando para abrir el editor SQL. El editor SQL no solamente sirve para generar o construir tablas, en general todos los objetos que existen en un modelo de BD se pueden generar desde el editor mediante la utilización de *Structured Query Language* o Lenguaje Estructurado de Consultas (SQL). SQL es el lenguaje más utilizado para el manejo de las BD Relacionales, además es el Estándar implementado en los SMBD Relacionales. Ha recibido el respaldo del *American National Standards Institute* (ANSI) como el lenguaje seleccionado para el manejo de BD Relacionales y es el lenguaje de acceso a datos que usan muchos productos SMBD comerciales como *DB2, SQL/DS, Oracle, INGRES, SYBASE, SQL Server, dBase* para *Windows, Paradox, Microsoft Access* y muchos otros (Kroenke, 2003).
5. Por último, es requerido ejecutar el código SQL para crear la estructura de la BD, se Ejecuta el código, automáticamente las tablas serán creadas y por consiguiente la BD estará lista para ser utilizada.

Cuando la base de datos está correctamente construida lo siguiente es iniciar con el paso número dos para construir la aplicación CRUD, el cual es la generación de las Entidades a partir de un diseño de BD. La generación de entidades no es un trabajo fácil de conseguir,

ya que las aplicaciones que hoy en día se desarrollan son Orientadas a Objetos, esto quiere decir que manejan Objetos y las bases de datos relacionales manejan tablas y registros; esta actividad de guardar objetos en una tabla no existe tal cual, para esto es necesario el uso de nuevas herramientas que cumplan con esta función, permitiendo que los objetos se puedan almacenar en tablas dentro de la base de datos.

Una herramienta muy poderosa y novedosa es el modelo de programación API de Java (JPA), el cual es una evolución y a la vez una recopilación de las características más eficientes de los modelos anteriormente utilizados (Panda et al., 2007), para realizar el mapeo de los objetos a las BD relacionales como *son Entity Beans 2.x , TopLink , Hibernate, JDO , y JDBC con DAO.*

JPA usa anotaciones para mapear objetos a la BD, estos objetos son llamados Entidades, las entidades JPA son clases POJOs, es decir son clases compuestas de código que no extienden de ninguna clase y no implementan ninguna interfaz. No es necesario manejar *Extensible Marckup Language* (XML) para hacer los mapeos. XML es un estándar para los lenguajes de marcas, un lenguaje de de marcas es un mecanismo para identificar estructuras en un documento (Rusty & Means, 2004).

Existen diferentes tipos de anotaciones que son utilizadas en la formulación de una Entidad, JPA maneja 4 tipos principales de anotaciones:

1. Las que definen que es nuestro objeto (**@Entity y @Embeded**).
2. Las que identifican al objeto en sí, para poder persistirlo (**@Id**).
3. Las anotaciones que se implementan para declarar relaciones entre diversos objetos (**@OneToOne, @OneToMany, @ManyToMany**).

4. Y por último las que definen cómo son los objetos que serán mapeados a la BD (**@Table, @Colum, @JoinColum**).

Las anotaciones son en realidad los obreros y la parte clave de JPA que realiza los mapeos de los objetos a la BD. Para generar las Entidades en *NetBeans* se utiliza el Asistente para generar Entidades desde una BD o "Generating Entities from DataBase", esta opción permite generar las Entidades correspondientes al proyecto en el que se está trabajando y principalmente basadas en las tablas que conforman a la base de datos, el IDE genera una entidad para cada una de las tablas que existen en el contexto de datos y las nombra igual que las mismas tablas, dentro de cada entidad existen propiedades. Estas propiedades las genera el IDE basándose en los campos de las tablas que existen en la BD y les asigna un nombre basado en cada columna de cada tabla en la BD. Incluso genera entidades para las tablas que son relacionadas entre sí, es para esto que se utilizan las anotaciones como **@OneToOne, @OneToMany**, que determinan en que grado una entidad se relaciona con otra.

El procedimiento para generar las Entidades, por medio del asistente para Generar Entidades desde una BD, es el que muestra en la Figura 4.

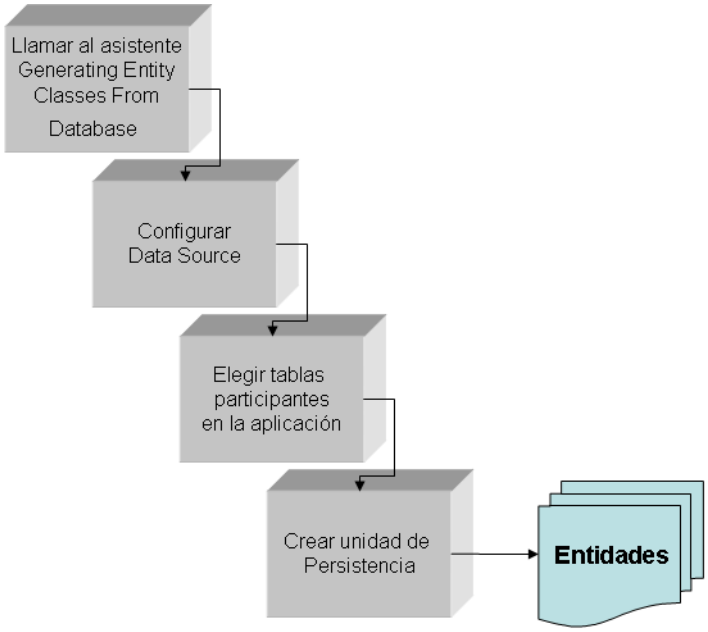

Figura 4. Procedimiento para la generación de Entidades

El procedimiento de la Figura 4 es muy general, con el fin de mostrar lo más relevante en la formulación de las entidades. Enseguida se enlista el procedimiento a mayor detalle:

1. En la ventana de "Proyectos" seleccionar dando "Click" derecho en el nodo principal del proyecto, para elegir la opción Nuevo > *Entity Classes From Database*, el cual es el asistente para la creación de la entidades de la aplicación.
2. Dentro del asistente, primero es necesario configurar el *DataSource* (DS), un DS o Fuente de Datos en español, es la representación para una fuente de datos (Sun Microsystems, 2001). Para elegir el DS que es requerido se elige del *combobox* en donde están contenidos los DS la opción Nuevo DS, después es necesario asignar un nombre al JNDI; un JNDI es una Interfaz de Programación de Aplicaciones (API) para servicios de directorio. Esto permite a los clientes descubrir y buscar objetos y nombres a través de un nombre. Y por último se elige la conexión a la BD de la aplicación.

3. Una vez que el DS fue configurado, lo siguiente es elegir las tablas que conformarán a la aplicación. Si todas son requeridas se elige la opción *Add All* y siguiente, esta opción depende del proyecto que se esté desarrollando.
4. Ahora se asigna un nombre al paquete de entidades, y se asegura que la opción *Generate named Querys* esté seleccionada.
5. Se elige la opción "Crear unidad de persistencia" para abrir el asistente de creación, se selecciona "Crear" en el asistente para establecer la unión de persistencia y se regresa al asistente para Generar Entidades desde la BD.
6. Se selecciona "Terminar" y automáticamente las entidades han sido creadas.

Durante el proceso de creación de entidades el IDE examina las relaciones entre las tablas, ya que genera entidades para todas las tablas, menos para tablas que son la unión de dos tablas. El IDE también genera entidades para las tablas que están compuestas de llaves primarias compuestas, las llaves primarias compuestas son llaves primarias que fueron designadas por el motivo de que existe una relación entre dos o más tablas.

Una vez que las entidades fueron desarrolladas, lo siguiente es el paso que número 3 para el desarrollo de la aplicación CRUD, el paso es Crear Páginas JSF desde las entidades o bien *Generate JSF pages from entity classes*. Para esto utilizaremos la tecnología *Java Server Faces* (JSF), JSF es un *Framework* o marco de trabajo que es basado en el Patrón de Diseño Modelo-Vista-Controlador (MVC), este patrón principalmente divide toda la aplicación en 3 componentes o espacios que trabajan de manera independiente:

- El Modelo, compuesto por las Entidades.
- La Vista, compuesta por los JSP o las páginas HTML que el usuario visualiza en su interacción con la aplicación.

- Y el Controlador, que es una clase que contiene un numero indefino de métodos que sirven a la aplicación y es quien administra todo lo que se realiza entre los diferentes componentes de la misma.

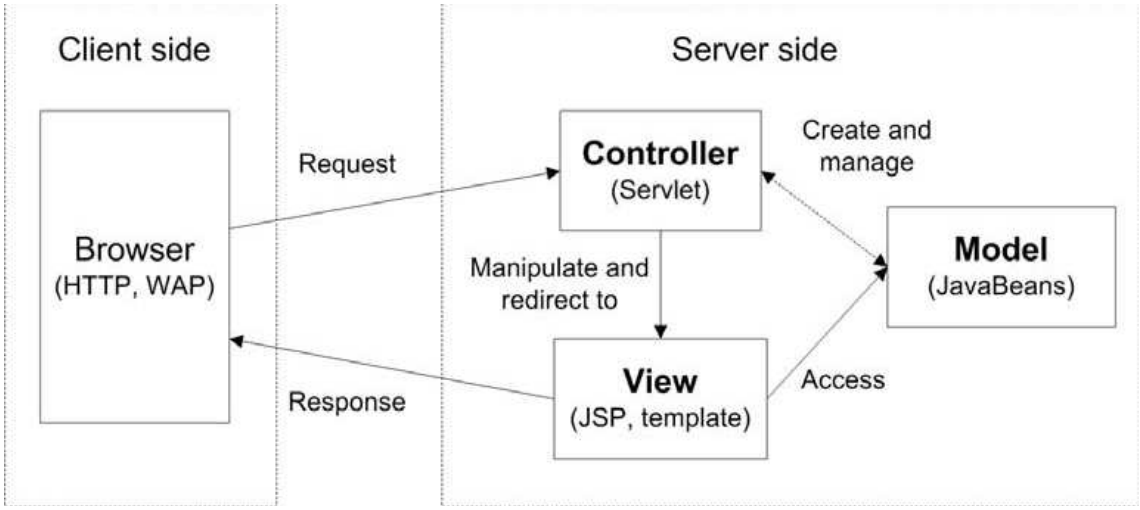

Figura 5. Modelo MVC

La figura 5 muestra un modelo que explica cómo funciona el Modelo MVC, la forma como se comunican todos los componentes que conforman a la aplicación, y también el lado del cliente que interactúa con la aplicación. En la primera sección de la figura 5 (en la parte izquierda) se muestra el lado del cliente, básicamente que el cliente hace una solicitud a la aplicación por medio de un explorador de Internet, esta solicitud la recibe el control que a su vez le muestra vistas al cliente según sea su solicitud. En el modelo MVC no todos los componentes saben que los otros componentes existen, por ejemplo el control sabe que existen la Vista y el Modelo, a diferencia de la Vista que no sabe que existe el Modelo, ésta tiene interacción directa con el controlador. Y el Modelo por su parte sólo sabe que existe un control, más no sabe nada de la Vista.

Más claramente, el funcionamiento del Modelo MVC se presenta por ejemplo en una aplicación para agregar un Alumno a una universidad, en este caso los eventos que se presentarían son los siguientes:

1. El cliente hace la solicitud para agregar el alumno.
2. El control manipula la vista, para que ésta muestre una pantalla en donde se introducen los datos del alumno.
3. Una vez introducidos los datos, el control solicita al modelo para que esté en base a lo que el cliente solicita y lleve a cabo las operaciones pertinentes, en este caso se ejecuta la operación **New()** para agregar aun nuevo Alumno.
4. El modelo ejecuta y el control le indica a la vista que muestre un mensaje de alumno agregado. Así, los tres componentes trabajan de manera independiente sobre un mismo proceso y dan solución a las necesidades del usuario.

JSF trabaja bajo este modelo o patrón de Diseño para ejecutar las peticiones del cliente del lado del servidor y mostrarle resultados del lado del cliente. En el desarrollo de aplicaciones, JSF permite la generación de las vistas, a las cuales el cliente tendrá acceso para mostrar y modificar información (Mann, 2004). Para la generación de las páginas JSF, *NetBeans* cuenta con el Asistente para generarlas desde las entidades, al igual que el asistente para generar las Entidades desde las tablas de la BD; este asistente es muy fácil de utilizar y configurar. Los pasos principales se muestran en la Figura 6, la figura define las actividades que se tienen que ejecutar para la generación de las páginas JSF.

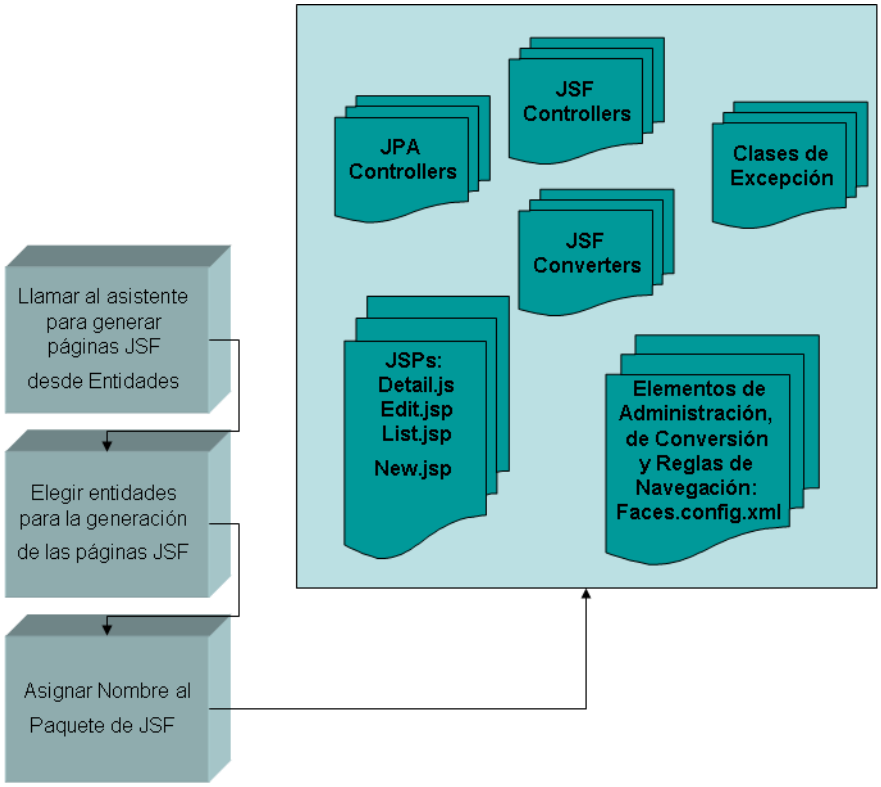

Figura 6. Proceso para la generación de páginas JSF

1. "Click" derecho sobre el Nodo del proyecto y elegir Nuevo > Asistente para generar páginas JSF desde Entidades, este asistente es el encargado de crear la parte de las vistas de la aplicación.
2. El asistente muestra una lista de todas las Entidades que tiene el proyecto hasta ese punto, es de esas entidades de donde se eligen cuáles se complementarán con páginas JSF, si es requerido (como en este caso) se elige la opción Add> Siguiente.
3. Ahora el asistente pide asignar un nombre para el paquete, se asigna y finaliza el asistente.

Al Finalizar el asistente el IDE genera las páginas JSF, para cada entidad el Asistente crea lo siguiente:

1. Una clase **JPA Controller**, este tipo de clases son las encargadas de manejar las operaciones que corresponden a cada Entidad, incluyendo las instancias de creación, editado y eliminación de la misma, así como el manejo de excepciones.
2. Una clase **JSF Controller**, cada una de estas clases está específicamente diseñada para cada JSP que JSF genera, estas clases incluyen código que invoca métodos en los JPA Controllers y determina cuales excepciones corresponden.
3. Una clase **JSF Converter**, cada una de estas clases implementa la interfaz *javax.faces.convert.Converter* definida por JSF, estas clases realizan trabajos para convertir instancias de una determina Entidad a objetos *String* y viceversa.
4. Un directorio que contiene 4 páginas JSP (Detail.jsp, Edit.jsp, List.jsp, New.jsp), que son las funcionalidades en sí de la aplicación CRUD, la tecnología JSP permite el desarrollo rápido de aplicaciones basadas en Web que son independientes de la plataforma. La tecnología JSP separa la interfaz de usuario de la generación de contenidos, permitiendo a los diseñadores a cambiar el diseño de la página en general, sin alterar el contenido dinámico subyacente.
5. Elementos de Administración, elementos de Conversión y reglas de navegación para la clase; las reglas de navegación son contenidas en un archivo XML (*faces-config.xml*), el cual se encuentra en la carpeta "*configuration Files*" dentro de la Solución.
6. El asistente también genera clases de excepción usadas por las clases JPA Controller.
7. Clases de utilidades usadas por las clases JSF Controller.

Al término de la creación de las clases JSF la aplicación está completa, pero solamente hasta la parte funcional de un CRUD. Como ya se había mencionado, el alcance de este proceso abarca hasta el punto en donde se consigue una aplicación totalmente funcional que guarda, elimina, consulta y modifica registros en una BD y que cuenta con las diferentes JSP para navegar dentro de la aplicación Web. Si se observa detenidamente, la aplicación se crea automáticamente. El desarrollador invirtió más trabajo (por así decirlo) en la parte de la generación de la BD, porque los asistentes de *NetBeans* prácticamente hacen todo lo demás. Ahora bien, una aplicación Web tiene que contar por lo menos con una parte de autentificación de usuarios, con un diseño de interfaz amigable, entre otras cosas, y es recomendable la implementación de estas características en la aplicación.

En conclusión, se puede afirmar que la metodología sí funciona, permite la generación de una aplicación CRUD de manera rápida, fácil y además trabaja bajo modelos de programación novedosos, basados en herramientas funcionales, en estándares conocidos y probados. Es conveniente, porque hoy en día el desarrollo de aplicaciones, indiferentemente del tipo que sean, son proyectos un tanto tardados por razones como que el desarrollo del código lleva tiempo porque se realizará desde un inicio y esto implica pruebas y correcciones, ya que pasar del diseño a la implementación no es una actividad fácil, o porque no se tiene al personal que conoce las herramientas que se requieren. Caballero (2006) afirma que los proyectos de software se encuentran pobremente administrados. Frecuentemente se retrasan o sobrepasan lo presupuestado inicialmente (se estima un factor del 50 al 100%), además de que los clientes o usuarios de la misma manera se muestran insatisfechos con la calidad de los sistemas de software. Es por esto que no es de sorprender que las organizaciones de desarrollo de software busquen activamente nuevas maneras de mejorar su desempeño.

Una alternativa de mejora es precisamente esta metodología de desarrollo, porque es fácil de usar e incluso el personal que no conoce mucho de desarrollo con Java puede hacerlo fácilmente y familiarizarse con la herramienta de manera inmediata. Es tan funcional que permite agilizar el desarrollo de aplicaciones, y si éste se hace en un menor tiempo lógicamente el costo de producción también se reducirá. Se fabricarán aplicaciones seguras, ya que esta herramienta toma en cuenta todos los posibles errores y validaciones de acuerdo a las reglas establecidas en la BD y a la integridad referencial de la misma, cosa que no se asegura en aplicaciones en donde el código se realiza "a mano", método por método. Esta metodología es una muy buena opción para desarrolladores de aplicaciones Web, totalmente recomendada y funcionalmente hablando, se podría decir que sin darse cuenta el desarrollador genera una aplicación CRUD totalmente funcional, con la inversión de muy poco tiempo de producción y esfuerzo.

Se sabe también que hoy en día las organizaciones que desarrollan software utilizan en su mayoría herramientas con licenciamiento, es normal y conveniente que si se trabaja con diferentes proyectos y clientes las empresas cuenten con un respaldo de una licencia en algún momento de problemática. Pero, por otra parte, si se trata de .NET es posible decir que es lo mismo que Java, ya que parten del mismo desarrollador. La diferencia radica en que Java es libre distribución y .NET no. Depende de cada empresa o persona la herramienta que desee utilizar, sólo que la funcionalidad será la misma con las diferentes variaciones de la plataforma y el lenguaje de programación que se use. Finalmente, la herramienta cuenta con muchas ventajas y es recomendada para cualquier desarrollo CRUD que se pretenda realizar rápida y eficazmente, orientada en un modelo relacional y no tanto en el modelo de objetos que conforma a una aplicación, ya que la base para su construcción es el modelo de relacional.

Referencias

Caballero, O. (Junio 2006). “Tecnologías de Información y herramientas para la administración de proyectos de software”. Revista Digital Universitaria. (Ver http://www.revista.unam.mx/vol.7/num6/art47/int47.htm).

De Pablos, C., López-Hermoso, S., Martin-Romo, S. & Medina, S. (2004). Informática y Comunicaciones en la Empresa, 186-189. Primera edición. ESIC Editorial: España.

INEGI (2003). Establecimientos que usan tecnologías de la información en sus procesos y relaciones con los clientes, por entidad federativa. (Ver http://www.inegi.org.mx/ est/contenidos/espanol/rutinas/ept.asp?t=apin79&s=est&c=14151).

Kroenke, D. (2003). Procesamiento de Base de Datos, 211-231.Octava Edición. Pearson Educación: México.

Lackerbauer, I. (2001). Internet, 11-15. Primera Edición: Alfaomega: España.

Mann, K. (2004). Java Server Faces in action, 38-56. Primera edición. Manning Publications Co.: Estados Unidos.

Myatt, A. (2007). Pro NetBeans IDE 5.5 Enterprise Edition, 45-60. Primera edición. Apress: Estados Unidos.

NetBeans Web Site (s.f.). NetBeans IDE-Connecting Developers. (Ver http://www.netbeans.org/ features/)

Rusty, E. & Means, W.S. (2004). XML in a nutshell: a desktop quick reference, 5-11 . Tercera Edición. O´Reilly: Estados Unidos.

Panda, D., Rahman, R. & Lin, D. (2007). EJB 3 in action, 250-291.Primera edición. Manning Publications Co.: Estados Unidos.

Sun Microsystems. (2001). Data Source. *(Ver* http://java.sun.com/j2se/1.4.2/docs/ guide/jdbc/getstart/datasource.html*)*