2. Antecedentes

2.1 Tecnologías del lado del cliente (Client-Side Technologies)

2.1.1 Lenguaje de hipertexto (HTML)

Una página web es el elemento básico para el desligue de información en ***Internet,*** generalmente está compuesta de lenguaje de hipertexto y la información del autor. El lenguaje de hipertexto es interpretado por una aplicación especial llamada ***navegador*** o explorador web***,*** el cual puede hacer un despliegue de la información que se desea mostrar.

El lenguaje ***HTML*** (conocido así por sus siglas en inglés: **H**yper**T**ext **M**arkup **L**anguage), basa su estructura en *elementos* cuya función es *describir* el formato con el que la información se mostrará en el navegador***.*** Cada elemento HTML tiene dos propiedades básicas: los atributos y el contenido; la función de los atributos es configurar el elemento creado y el contenido puede ser texto plano o inclusive otros elementos HTML.

La sintaxis de HTML está basada en etiquetas, cada vez que es escrita una **etiqueta HTML,** es creado un elemento, cada etiqueta está delimitada por los símbolos “< >” y “</ >”. El símbolo “< >” indica la apertura o inicio de una etiqueta HTML, mientras que “</>” indica el fin de la misma, cada elemento posee un nombre o abreviatura que debe ser escrito dentro en las etiquetas de inicio y fin.

El lenguaje HTML es una herramienta poderosa para lograr la presentación de información de forma adecuada, sin embargo, el costo en horas de codificación de página y corrección de errores es alto. El trabajo de escritura de páginas web a través de código HTML puro resulta, tardado, laborioso y no contribuye a la creación de páginas dinámicas o con presentaciones exigentes ya que se necesita codificar bastantes etiquetas para que toda la página conserve una coherencia en su presentación, es por ello que existen herramientas que facilitan entre otras cosas la de crear la presentación de las páginas web, con la ventaja de separar la información, la funcionalidad y la presentación del documento.

Actualmente HTML está migrando a **(x)HTML** (acrónimo en inglés de e**X**tensible e**H**yper**t**ext **M**arkup **L**anguage), este lenguaje de marcado de hipertexto Extensible es una versión más estricta y limpia de HTML. (x)HTML extiende HTML 4.0 combinando la sintaxis de HTML, diseñado para mostrar datos, con la de XML, diseñado para describir los datos. (xhtml,2010).

2.1.2 Hojas de estilo en cascada

Como se observó en el apartado anterior, la creación de una página con HTML puede llegar a ser un trabajo arduo y complicado, esto sin mencionar que su código será muy extenso por lo que se dificulta la corrección de errores de información y de diseño. Para evitar todo tipo de conflictos en el desarrollo de una página web se han creado herramientas que facilitan a los programadores proveer a las páginas una presentación sin necesidad de estar totalmente codificada en HTML, entre ellas se encuentran las hojas de estilo en cascada.

Las hojas de estilo en cascada, conocidas también como **CSS** (por su abreviatura en idioma inglés : **C**ascading **S**tyle **S**heets), son un lenguaje de hoja de estilo para

la Red Global Mundial o World Wide Web (WWW). Éste describe la presentación (por ejemplo, fuentes, colores y espacios) de documentos estructurados. CSS puede ser escrito y leído por los humanos y expresa en terminología, el estilo de una publicación (CSS, 1998). CSS se basa en estilos, los estilos definen cómo se despliegan los elementos HTML en el navegador.

La escritura de una página HTML con CSS es bastante organizada, ya que CSS proporciona reglas muy sencillas -pero muy poderosas- para dar presentación a las páginas. Estas reglas se basan en un ***paradigma orientado a objetos*** (POO), por ejemplo, cualquier elemento HTML puede pertenecer a una clase y ésta le otorga todas sus propiedades configuradas.

Las definiciones de CSS pueden encontrarse dentro de la misma página HTML o bien se puede colocar en un archivo externo, con la única condición de que se ligue en la cabecera de la página.

2.1.3 Modelo de objetos del documento (DOM)

Abreviado DOM por su nombre en inglés (Document Object Model) es una plataforma -y un lenguaje- de interfaz neutral que permite a los programas y scripts acceder y actualizar dinámicamente el contendido, la estructura y el estilo de los documentos HTML. El documento puede ser posteriormente procesado y los resultados de ese tratamiento pueden ser incorporados de nuevo en la presentación de la página. (DOM,2005)

Cada vez que es cargada una página web en el navegador se crea un árbol jerárquico de elementos (x)HTML, este árbol es conocido como el objeto DOM. Esta estructura de tipo árbol está compuesta por nodos, cada nodo es un

elemento (x)HTML representado en el árbol, cada nodo puede ser una rama que puede tener hijos y padres.

La manipulación de la estructura de árbol de cualquier página web es posible gracias a que DOM está provisto de una ***interfaz de programación de aplicación (API)***. Esta API posee métodos especiales para el acceso y modificación de los nodos inclusive la creación de nuevos. El nodo *document* por ejemplo, tiene entre sus métodos a getElementById() y setAppendChild(), el primero permite el acceso a un nodo y el segundo agrega un nuevo nodo a la estructura del árbol jerárquico.

La manipulación del DOM, entonces, hace posible que una página pueda modificar su contenido inclusive aun cuando un usuario esta interactuando con ella, esta funcionalidad hace que se puedan desarrollar páginas más robustas y dinámicas, aumentando favorablemente la experiencia que recibe un usuario al visitar dicha página.

2.1.4 Java Script

JavaScript es un lenguaje basado en LiveScript creado por Brendadn Eich para la empresa Netscape Communications, LiveScript hizo su primera aparición en Netscape Navigator 2.0 y más tarde en diciembre de 1995 fue rebautizado con el nombre de JavaScript por Sun Microsystems y Netscape (ECMA, 2009).

En 1996 se comenzó el desarrollo del estándar ECMA, el cual es un lenguaje (o motor) basado en lenguajes JavaScript y JScript de Microsoft. El estándar ECMA es soportado e interpretado por la mayoría de los ***navegadores*** modernos y se utiliza mayormente para realizar operaciones en la aplicación del cliente al mismo

tiempo que las sentencias van descargándose junto con el código HTML, JavaScript es ahora un ***dialecto*** de ECMA así que para fines prácticos de esta tesis el término ECMA y JavaScript se utilizarán de modo indistinto.

JavaScript puede realizar operaciones complejas dentro del navegador del cliente. Inclusive, si se le provee de una implementación adecuada, es capaz de modificar el DOM, por lo que es una forma efectiva y poderosa de proveer a las páginas de mayor dinamismo y evitar trabajos excesivos e innecesarios para el servidor. A continuación se muestra la pantalla de una aplicación basada en el juego de la vida y desarrollada en totalmente JavaScript.

Figura 2.1.1 Fragmento de página Web que contiene una aplicación basada en el juego de la vida e implementada totalmente en JavaScript.

2.1.5 AJAX

Este término fue utilizado por primera vez en febrero de 2005, por Jesse James Garrett en el artículo: "Ajax: A New Approach to Web Applications ", AJAX es un acrónimo de *Asynchronous JavaScript + XML* ("JavaScript asíncrono + XML) y define formalmente los nuevos tipos de aplicaciones web que se desarrollan hoy en día.

Definiendo Ajax: "Ajax no es una tecnología en sí mismo. En realidad, se trata de varias tecnologías independientes que se unen de formas nuevas y sorprendentes."

Las tecnologías que forman AJAX son:

- XHTML y CSS, para crear una presentación basada en estándares.
- DOM, para la interacción y manipulación dinámica de la presentación.
- XML, XSLT y JSON, para el intercambio y la manipulación de información.
- XMLHttpRequest, para el intercambio asíncrono de información.
- JavaScript, para unir todas las demás tecnologías. (Garrett, 2005)

El día de hoy, AJAX es esencial para el desarrollo de sistemas web competitivos, esto sin mencionar que gracias a su constitución permite mejorar el rendimiento de los sistemas con la ventaja de ofrecer páginas más dinámicas a los usuarios.

El objeto esencial del funcionamiento de AJAX es el *XMLHttpRequest()* creado por Alex Hopmann en el año 2000. Este objeto se encarga de realizar la comunicación asíncrona con el servidor, primero se instancia el objeto XMLHttpRequest, este objeto depende del navegador que lo solicite, por ejemplo Firefox, o Safari

generan el objeto XMLHttpRequest de forma nativa, por lo que se puede obtener a través del objeto window, de no ser así lo obtienen a través de un recurso extra conocido como *complemento* o *plug-in*.

Una vez que el objeto XMLHttpRequest es instanciado, éste prepara la función de respuesta y realiza la petición al servidor, el servidor ejecuta la petición y XMLHttpRequest se encarga de ejecutar la función de respuesta, a continuación se muestra el diagrama de comparación entre el modelo tradicional de web y el que ofrece Ajax:

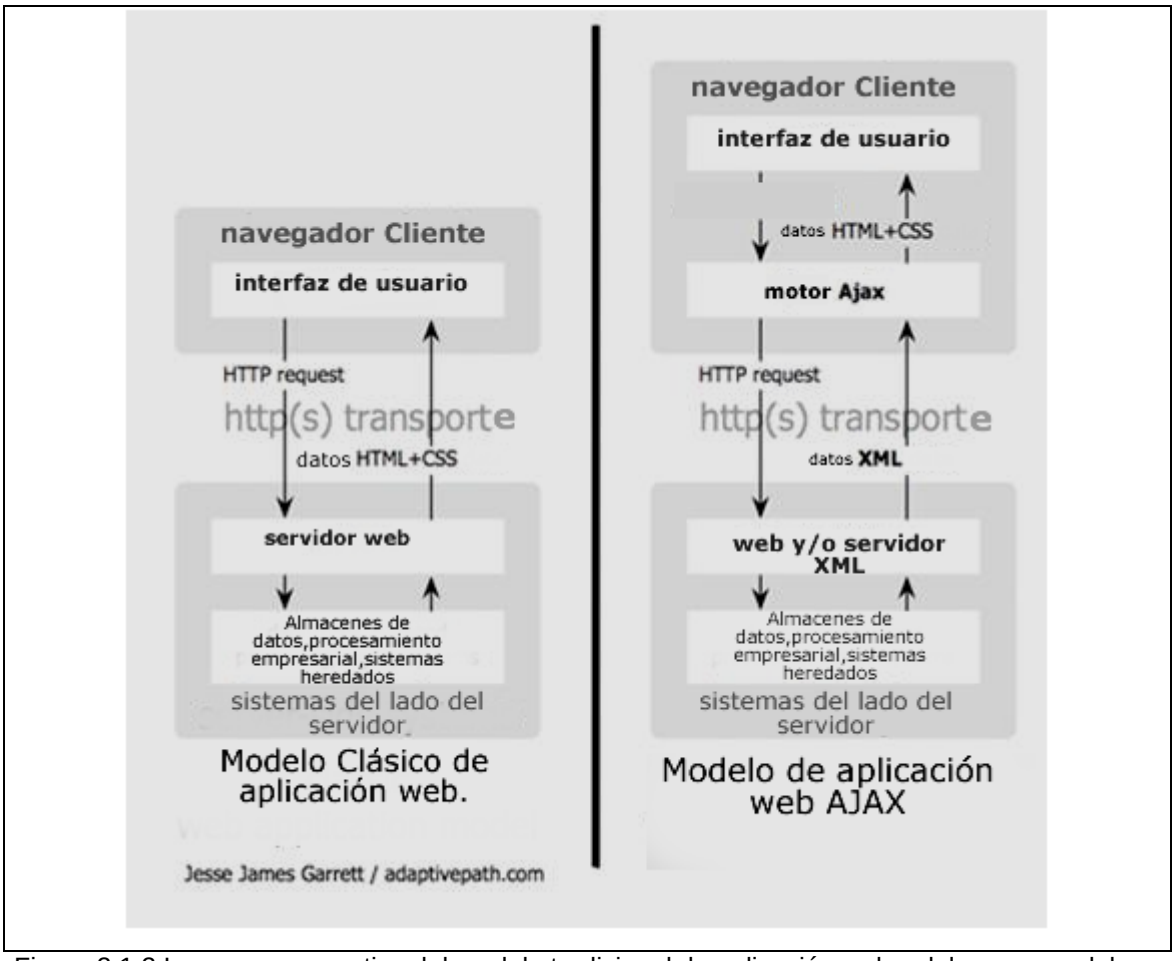


Figura 2.1.2 Imagen comparativa del modelo tradicional de aplicación web y del nuevo modelo propuesto por AJAX (Garrett,2005).

El ejemplo que se muestra a continuación es una pequeña página (Figura 2.1.3) en la que se agregan objetos utilizando el DOM (Figura 2.1.4) y al dar un click en el objetos creados se realiza una llamada Ajax (Figura 2.1.5), esta llamada se comunica con el servidor. El servidor sabe que cuando reciba una petición del objeto tiene que responder con un saludo, este saludo pasa por el motor de Ajax y se muestra en el navegador sin necesidad de recargar la página (Figura 2.1.6).

Figura 2.1.3 Ejemplo de una página que crea un objeto por medio de DOM.

Figura 2.1.4 Muestra un objeto creado por medio de DOM.

Figura 2.1.5 Al dar click en el objeto se hace una llamada AJAX.

Figura 2.1.6 Se muestra la respuesta del servidor en la página.

2.1.6 Librería Yahoo de interfaz de usuario (YUI)

YUI es la abreviatura de Yahoo User Interface. E*sta librería es un conjunto de utilidades y controles, escrita con JavaScript y CSS, para la construcción de aplicaciones web interactivas ricas utilizando técnicas como el scripting DOM, DHTML y AJAX. YUI está disponible bajo una licencia BSD y es gratuito para todos los usos*. (YUI, 2010)

YUI es continuamente probada, por lo que asegura ser escalable y robusta, esta librería es creada por los ingenieros de interfaz de Yahoo y colaboradores de todo el mundo. Finalmente, YUI se puede definir como una librería en JavaScript de potencia industrial, que ayuda a los desarrolladores Web a realizar sistemas más interactivos sin necesidad de empezar dicha interactividad desde cero.

2 Antecedentes

Hoy en día, YUI cuenta con tres versiones las cuales son: YUI, YUI 2, y por último YUI 3. Dichas versiones son el resultado de extensiones y mejoras de su antigua versión, YUI 3 por ejemplo, es la nueva generación de *JavaScript y CSS, YUI3 funciona en el correo oficial de Yahoo! e incorpora lo que los desarrolladores* han aprendido en los cinco años de producción de la biblioteca *(YUI3, 2010).*

A pesar que JavaScript pretende ser estándar, un mismo código de YUI puede llegar a tener comportamientos inesperados en algunos navegadores, por esta razón una vez que todo el código YUI tiene el comportamiento esperado sobre un navegador en especifico los desarrolladores de YUI lo clasifican como un navegador grado A (“ **A-grade Browser**”, consultar el glosario para mayor información).

La arquitectura base de YUI está concentrada en el archivo yui-core.js y en el YahooGlobalObject, ambos ofrecen todos los recursos necesarios para dejar a cualquier ***navegador grado A*** listo para la creación de elementos YUI, yui-core.jsp proporciona entre muchas otras cosas el mecanismo de carga de script y las funciones esenciales de toda la biblioteca YUI, mientras que el YAHOOGlobalObject proporciona un único namespace global en el que todo el código de la biblioteca YUI reside.

YAHOO! DEVELOPER NETWORK

YUI Components

Animation	Layout Manager
AutoComplete	Logger
Browser History Manager	Menu
Button	Paginator
Calendar	Profiler
Carousel BETA	ProfilerViewer
Charts BETA	ProgressBar BETA
Color Picker	Resize
Connection Manager	Rich Text Editor
Container	Selector
Cookie	Slider
DataSource	Storage BETA
DataTable	StyleSheet BETA
Dom	SWF BETA
Drag & Drop	SWFStore BETA
Element	TabView
Event	TreeView
Get	Uploader BETA
ImageCropper BETA	Yahoo Global Object
ImageLoader	YUI Loader
JSON	YUI Test
	Reset CSS
	Base CSS
	Fonts CSS
	Grids CSS

Tabla 2.1.1 Lista de Componentes de la librería .YUI

El objeto YAHOOGlobalObject es lo único que invariablemente debe estar incluido cuando se desee utilizar YUI, ya que no sólo crea los espacios de nombres que utilizan los otros elementos YUI, sino que también contiene funciones especiales que estos componentes YUI necesitan para trabajar.

YUI provee una gran gama de componentes (ver tabla 2.1.1), cada uno facilita a los desarrolladores web tareas detrás de la aplicación como JSON utility o el Connection Manager o bien tareas para mejorar la experiencia de interacción del usuario como la utilidad de calendario, la vista de árbol o arrastrar y soltar.

En la figura 2.1.7 se muestra el uso de la utilidad arrastrar y soltar que provee YUI, en unos cuantos pasos, gracias a la librería YUI se puede tener en el navegador

una interacción con objetos web moviéndolos de lugar en el explorador.

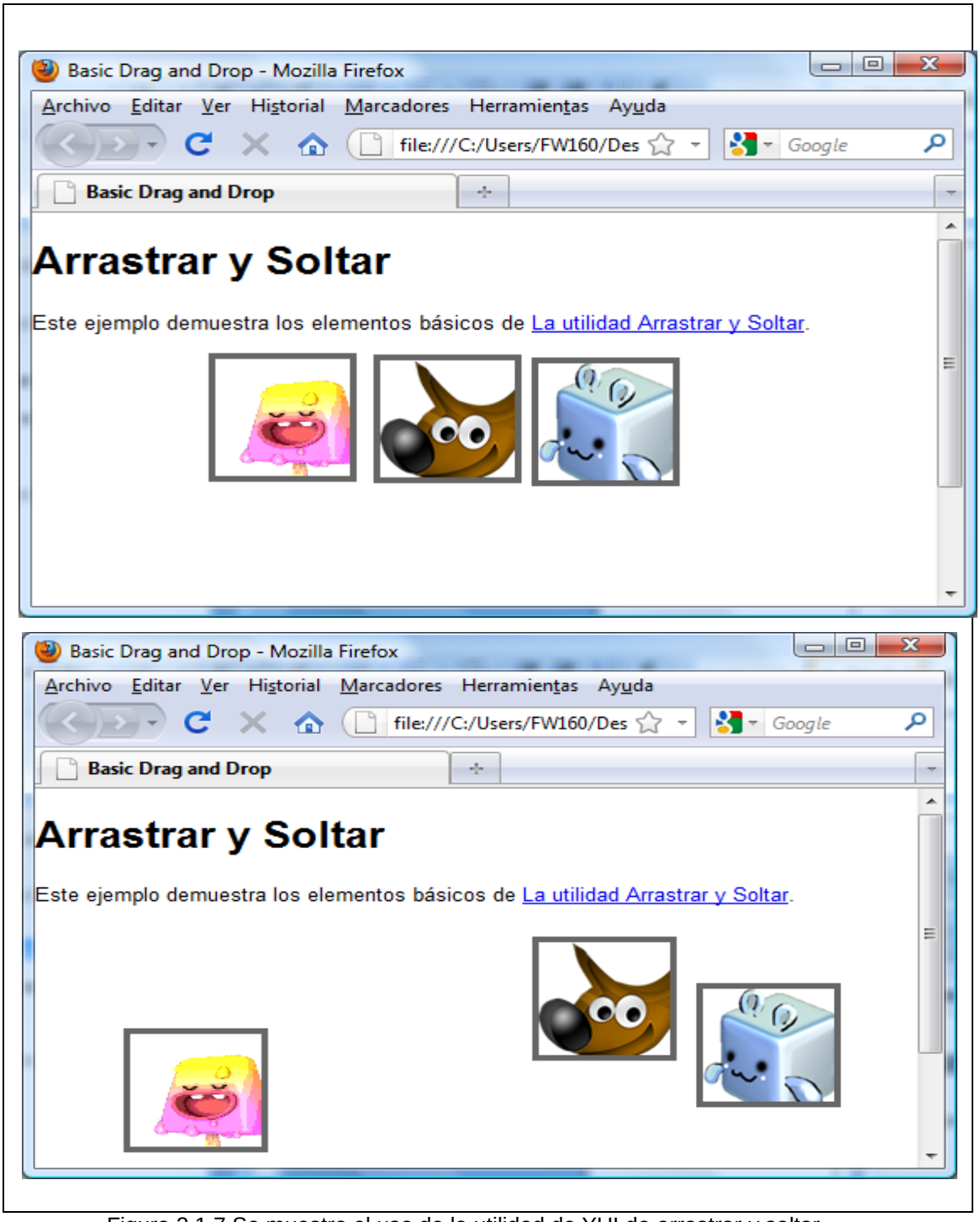


Figura 2.1.7 Se muestra el uso de la utilidad de YUI de arrastrar y soltar.

2.1.7 Flash

Adobe Flash es una aplicación multimedia usada para desarrollar animaciones, vídeo e interactividad para las páginas Web. **Adobe Flash** es muy usado en anuncios y juegos Web.

Adobe Flash trabaja sobre "fotogramas" y está destinado a la producción y entrega de contenido interactivo para las diferentes audiencias alrededor del mundo sin importar la plataforma ya que cuenta con su propia “máquina virtual” llamada **Adobe Flash Player**.

Adobe Flash soporta:

- ***gráficos vectoriales*** e ***imágenes raster***.
- sonido
- Programación en ***ActionScript 3.0***
- Flujo de vídeo y audio bidireccional

2.2 Tecnologías del lado del servidor (Server-Side Technologies)

2.2.1 Bases de datos

2.2.1.1 Introducción

Desde que las computadoras fueron capaces de guardar información fue necesario también crear métodos para que dicha información se almacenara de forma organizada. Durante los años cincuenta los archivos secuenciales, de

acceso directo e indexado eran las únicas maneras de tener acceso a la información almacenada, ya que el programa y los datos estaban completamente ligados.

A principios de la década de los sesenta se comenzó el uso de los discos flexibles en donde los accesos directos recuperaban la información esparcida y la consulta de la información no dependía directamente de las aplicaciones, a mediados de esta década los volúmenes de información ya eran considerablemente grandes, así que surgieron sistemas de almacenamiento basados en punteros físicos (direcciones de memoria que han referencia a un dato) conocidos como sistemas de bases de datos.

Los primeros sistemas de bases de datos se basaban en la estructura jerárquica de los datos, lo cual permitía recuperar múltiples registros asociados a un registro único de otro archivo, poco después surgieron los sistemas de bases de datos en red los cuales permitían interrelaciones más complejas entre registros de archivos.

En 1970 E.F.Codd publica un artículo sobre el modelo relacional de los datos (Codd, 1970), en el cual se propone el acceso y manipulación de los datos únicamente a través de sus características lógicas. Este artículo abrió nuevas expectativas en el campo de los sistemas de bases de datos y daría pie a los sistemas que vendrían en el futuro (Hansen et al, 1997).

2.2.1.2 Base de datos

Una base de datos es una colección de datos relacionados, es decir, información que puede ser guardada y tiene un significado implícito, una base de

datos tiene las siguientes propiedades implícitas:

Una base de datos representa algún aspecto del mundo real, algunas veces llamado minimundo o universo en discusión (DoD), los cambios en el minimundo son reflejados en la base de datos.

Una base de datos es una colección lógicamente coherente de datos con un significado inherente. Un surtido aleatorio de datos no puede ser contemplado correctamente como una base de datos.

Una base de datos es diseñada, construida y llenada con un propósito específico. Ésta tiene un conjunto de usuarios de propósito y algunas aplicaciones preconcebidas en la que dichos usuarios están interesados. (Elmasri, 2004)

Una base es un conjunto de datos almacenados entre los que existen relaciones lógicas y ha sido diseñado para saber los requerimientos de información de una empresa u organización. (SBD, 2007)

Las bases de datos como ya se mencionó anteriormente, existen por la necesidad de un conjunto de personas en específico, las cuales son las que le imprimen importancia a la información almacenada.

A las bases de datos se les puede clasificar en dos grandes grupos: las estáticas y dinámicas. Las estáticas son de sólo lectura y son regularmente utilizadas para guardar datos históricos, las dinámicas por su parte se caracterizan por que su información va cambiando con el tiempo y permite operaciones particulares sobre los datos como son: Agregar datos, Borrar datos y Actualizar datos. (TDB, 2007)

Las bases de datos traen consigo muchas ventajas como para los desarrolladores

de sistemas entre las que podemos mencionar las siguientes:

1.-Independencia de datos y tratamiento.- Los cambios en la información de las bases de datos no implican ningún cambio en las aplicaciones que tratan dichos datos.

2.-Mejora en la disponibilidad de datos.

3.-Coherencia de resultados.- Disminuye la redundancia y evita la inconsistencia.

4.-Cumplimiento de normas.- Ayuda a los involucrados a cumplir con las reglas de negocio.

5.-Seguridad.- Evita que personas no autorizadas manipulen los datos.

2.2.1.3 Arquitectura de las Bases de datos

En 1975 el comité ***ANSI-SPARC*** propuso una arquitectura de tres niveles para las bases de datos que permite separar la vista que los usuarios tienen de la base de datos (DB) de la forma en que es representada físicamente dicha base, en la tabla de define cada nivel propuesto por el estándar.

Nivel	Descripción
Nivel Externo (Vistas de usuario):	Es la vista de un usuario de la base de datos, esta vista muestra partes relevantes para un usuario en particular. Se excluyen los datos irrelevantes, así como los datos que el usuario no está autorizado a acceder.
Nivel Conceptual	El nivel conceptual es una forma de describir la forma en la que los datos se almacenan en la base de datos completa y cómo los datos son relacionados entre sí. El nivel conceptual no especifica cómo los datos se almacenan físicamente.
Nivel Interno	Es el modo en el que la base de datos está físicamente representada en el sistema informático. En él se describe cómo los datos se almacenan en la base de datos en hardware del equipo.

Tabla 2.2.1 Niveles de las bases de datos.

Cada nivel tiene su correspondiente esquema:

Esquema **nivel interno**.- Este esquema es el más bajo y contiene las definiciones de los registros almacenados, los métodos de representación, los campos de datos e índices. Sólo hay un esquema interno por base de datos. En este nivel se describe la estructura física de la base de datos y define los métodos de acceso.
Esquema **nivel conceptual**.- En este esquema se describen todos los elementos de datos y relaciones entre ellos, junto con las restricciones de integridad. Sólo hay un esquema conceptual por base de datos.
Esquema **nivel externo**.- El esquema externo muestra diferentes vistas de una base de datos, puede haber muchos esquemas externos para una base de datos dada.

Las ventajas de utilizar esta arquitectura son variadas, una es que hace que las bases de datos puedan personalizar las vistas del usuario de forma independiente, cada usuario debe poder tener acceso a los mismos datos, pero con un punto de vista diferente (personalización).

Otra ventaja es que se ocultan a los usuarios los detalles físicos de almacenamiento, los usuarios no deben tener que lidiar con los detalles físicos de almacenamiento aún cuando haya cambio el medio de almacenamiento.

Por último pero no por ello menos relevantes es que si un administrador hace cambios en la estructura conceptual de la base de datos, las vistas de los usuarios se mantendrán intactas.

2.2.1.4 Sistemas de base de datos

En cualquier organización un sistema de base de datos está formado por cuatro elementos esenciales: el hardware, el software, los datos y las personas.

2.2.1.4.1 El hardware

Es el conjunto de dispositivos físicos sobre los cuales es posible el manejo de la base de datos, entre esos dispositivos podemos mencionar las unidades de disco, una o más computadoras, etc.

2.2.1.4.2 El software

El manejo de la información residida dentro de cualquier unidad de almacenamiento sería imposible sin programas que sirvan de intermediario entre los datos y los usuarios, cualquier sistema de base de datos, por ejemplo, posee los siguientes tipos de software: El software de propósito general comúnmente llamado Sistema de Gestión de Base de Datos (SGBD) y el software de aplicación el cual usa las facilidades que le provee el SGBD para manipular las bases de datos con el fin de llevar a cabo una función como por ejemplo una consulta.

El Sistema de Gestión de Base de Datos puede ser un compilador o inclusive un **sistema operativo**, su trabajo es proveer a los usuarios o programadores de herramientas o servicios que faciliten la gestión de una base de datos, dichos servicios son los siguientes:

- Mecanismos de seguridad e integridad de base de datos.
- Diccionario o directorio de datos.

- Acceso concurrente a los datos para varios usuarios.
- Herramientas para la consulta, elaboración y manipulación de informes orientados al usuario.
- Herramientas para el desarrollo de sistemas de aplicación orientados al programador.

Dentro de un SGBD existen tres subsistemas cuyas funciones son definir, manipular y mantener la seguridad de los datos con el fin de proveer los servicios anteriormente mencionados:

DDL.- Es un lenguaje especial de definición de datos, (en inglés, data definition lenguage), dicho lenguaje se utiliza para crear los esquemas de definición de las DB. Cuando se compilan y procesan dichos esquemas se obtiene como resultado una serie de tablas que se almacenan en un archivo especial, dicho archivo es conocido como el **Diccionario o directorio de datos (DD/D).**

Diccionario o directorio de datos.	
Tipo de almacenamiento	Descripción
Nivel primario	Almacena los elementos de los datos, conocidos como Campos.
Nivel de estructura de datos	Almacena los datos a nivel de grupo, registro, tablas relacionales y archivos.

Tabla 2.2.2 Tipos de almacenamiento del diccionario de datos.

DML.- De sus siglas en inglés data manipulation language, es un lenguaje de manejo de datos, el cual permite a los usuarios manejar o tener acceso a datos

que fueron organizados a partir de un modelo apropiado, una parte muy importante de un DML es que éste realiza la recuperación de los datos. Esta parte es conocida como lenguaje de consultas.

DCL.- Es un lenguaje de control (en inglés, Data Control Language) que se encarga de administrar y plasmar los privilegios de los usuarios.

Tipo de DML	Descripción
De Procedimientos	En este tipo de DML el usuario especifica qué datos quiere y cómo deben obtenerse.
Sin Procedimientos	En este tipo de DML se requiere que el usuario especifique que datos desea sin especificar cómo deben obtenerse.

Tabla 2.2.3 Tipos de DML.

2.2.1.4.3 Los datos

Es esencial considerar que una base de datos está conformada por datos, dichos datos deben ser cuidadosa y lógicamente estructurados, además que las funciones de aplicación, los elementos de los datos, las interrelaciones y definiciones deben identificarse y definirse justamente antes de ser almacenados dentro del diccionario de datos. Una vez almacenados los datos podrán obtenerse e introducirse en la base de datos según la estructura cuidadosamente definida.

2.2.1.4.4 Las personas

Las personas que son encargadas de los procedimientos para lograr los objetivos de un sistema son esenciales ya que sin ellas la base de datos carecería de: organización, centralización e inclusive de información necesaria. Las personas pueden clasificarse en dos grandes grupos: los usuarios, y los profesionales en computación (ver tabla). (Hansen,et al,1997).

Tipos de personas	
Tipo	Descripción
Usuarios 	Son aquellas personas que necesitan la información de la base de datos para desarrollar sus responsabilidades en la institución.
Profesionales en computación 	Es responsables base de datos y de los paquetes relacionadas a ellos.
Administrador de la base de datos. 	Es aquella persona que tiene el control centralizado del SGBD, entre sus responsabilidades se encuentran: 1)Definir el esquema de la base de datos 2) Definir la estructura y método de almacenamiento. 3) Modificación de esquema y de la organización física. 4)Concesión de autorización para acceso a los datos 5)Especificación de las limitantes de integridad

Figura 2.2.1 Tipos de personas.

2.2.2 Tecnología Java

Desarrollada por ***Sun MicroSystems*** la tecnología Java puede ser vista como un ambiente de desarrollo la cual provee herramientas básicas para este mismo fin, dichas herramientas son: un compilador, un intérprete, un generador de documentación y una serie de paquetes que contienen todas las clases necesarias para realizar todo tipo de aplicaciones.

Versiones de la Tecnología Java	
Java SE Java Estándar Edition (J2SDK) Estándar Development Kit	Se utiliza para el desarrollo y despliegue de aplicaciones Java en computadoras personales y servidores, así como los exigentes ambientes embebidos y en tiempo real.
Java Platform, Enterprise Edition (Java EE)	Es el estándar de la industria de la computación empresarial Java. Se utiliza para crear aplicaciones web de próxima generación, aplicaciones empresariales. Dichas aplicaciones son basadas en servlets, Java Server Pages y Enterprise Java Beans.
Java Platform, Micro Edition (Java ME)	Proporciona un entorno robusto y flexible para las aplicaciones que se ejecutan en móviles y otros dispositivos embebidos de teléfonos móviles, asistentes digitales personales (PDA), adaptadores de televisión, e impresoras. Java ME incluye interfaces de usuario flexibles, robustas funcionalidades de seguridad, una función de protocolos de red y soporte para aplicaciones de red y en línea que pueden ser descargados de forma dinámica.

Tabla 2.2.4 Versiones de Java (Java, 2010)

2.2.2.1 Tipos de aplicaciones Java

Como se puede ver en la tabla 2.2.4, la tecnología Java nos ayuda a generar una gran gama de aplicaciones entre las que podemos mencionar:

- Las aplicaciones.- Programas convencionales que se ejecutan bajo el control de un Sistema Operativo.
- Applets.- Programas que se ejecutan en un navegador WEB. (Explorer, Netscape).
- Servlets.- Aplicaciones que se ejecutan en un servidor de aplicaciones.
- Java Beans.- Componentes (generalmente gráficos) que siguen una serie de convenciones preestablecidas.

JSPs. (Java Server Pages).- JSP son elementos web convertidos en servlets por un servidor de aplicaciones, jsp separa la interfaz de usuario de la generación de contenidos (generalmente parte gráfica).
EJBs.-Aplicaciones que se ejecutan en un servidor de aplicaciones y se encargan de resolver la lógica empresarial.

B 2.2.2.2 El lenguaje Java

Java puede ser considerado también como un lenguaje de programación de alto nivel, dicho lenguaje es ***orientado a objetos,*** soporta ***multithreading*** y está caracterizado porque tanto su código fuente y binario son independientes de plataforma. El código binario por ejemplo, es capaz de funcionar en la mayoría de las arquitecturas conocidas (Intel, Sparc, Motorola, etc.), siempre y cuando el sistema operativo tenga instalado el intérprete de java.

2 Antecedentes

El lenguaje java es:

- **Sencillo:** Java posee una elegante sintaxis, y consta de datos primitivos, clases eliminando ***apuntadores*** y herencia múltiple.

- **Distribuido:** Java permite la construcción de aplicaciones distribuidas por medio de una colección específica de clases.

- **Robusto:** Java es un lenguaje robusto y fiable, se ha escrito pensando en poder verificar errores y está muy tipificado.

- **Seguro:** Java tiene escasos problemas de seguridad, característica muy importante en las aplicaciones distribuidas por internet.

- **Portable:** Java es un lenguaje de alto nivel e independiente de la plataforma, esto lo hace portable.

- **Alto rendimiento:** Los nuevos compiladores conocidos como JIT permiten un rendimiento muy parecido a los lenguajes convencionales compilados.

- **Dinámico:** En tiempo de ejecución, el entorno Java se puede extender mediante enlaces a clases que pueden estar localizadas en servidores remotos o en una red.

2.2.2.3 La plataforma Java

Una plataforma es el hardware o el entorno de software en el que se ejecuta un programa, algunas de las plataformas más conocidas son Microsoft Windows, Linux, Solaris y Mac OS. La mayoría de las plataformas se puede describir como una combinación del sistema operativo y el hardware subyacente. La plataforma Java difiere de la mayoría de las plataformas de otros en que es una plataforma de software, que corre por encima de otras plataformas basadas en hardware. (LJava ,2010).

- La plataforma Java está formada por dos componentes:
  - La máquina virtual de Java
  - La interfaz de aplicación de programación Java (API)

2.2.2.4 El API de Java

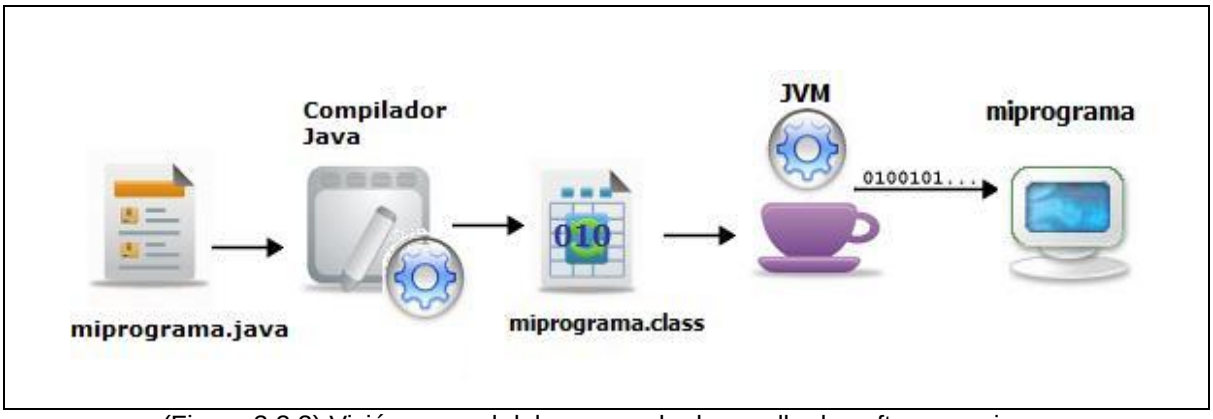
El API es una gran colección de componentes de software pre-elaborados que ofrecen muchas funciones útiles. Se agrupan en las bibliotecas de clases e interfaces conexos; estas bibliotecas se conocen como paquetes.

2.2.2.5 Compilador java (CJ)

El compilador de java es el encargado de revisar la sintaxis y la semántica del código fuente escrito en java, además de traducir dicho código a lenguaje máquina (un código en bytes), el interprete de java será capaz de ejecutar dicho código máquina.

2.2.2.6 La máquina virtual de Java (JVM)

La maquina virtual de java (JVM) es capaz de interpretar y ejecutar el código máquina (un código en bytes) generado por el CJ. El código que ejecuta la máquina virtual se encuentra en archivos .class que son el resultado de la compilación de los archivos fuente .java, los archivos .class contienen “byte codes” que son interpretados por una máquina específica para cada plataforma (Windows Microsoft, Linux, Mac OS). (Figura 2.2.2), esto podría significar un pequeño descenso en la eficiencia de java, sin embargo se ha comprobado que la máquina virtual interpreta y ejecuta a excelentes velocidades.



(Figura 2.2.2) Visión general del proceso de desarrollo de software en java.

2.2.3 Java Enterprise Edition

Java EE es una arquitectura que se utiliza para implementar aplicaciones empresariales utilizando Java e Internet, las tecnologías básicas de JEE son los Servlets, las Java Sever Pages (JSPs), y los Enterprise Java Beans (EJBs).

2.2.3.1 Aplicaciones en JEE

La plataforma Java EE utiliza para las aplicaciones empresariales un modelo de aplicación distribuida de varios niveles. La lógica de aplicación se divide en los componentes según su función y en los componentes que permiten que la aplicación java EE sea instalada en máquinas diferentes dependiendo de a qué nivel de entre todos pertenece en el entorno Java EE.

Los niveles en los que se dividen las aplicaciones JavaEE (app.JEE) son:

- Nivel cliente.- Componentes que se ejecutan en la máquina del cliente.
- Nivel Web.- Componentes que se ejecutan en el servidor Java EE.
- Nivel de Negocio.- Componentes que se ejecutan en el servidor Java EE.
- Nivel de Sistema de Información Empresarial (EIS-tier).-Es todo el software que se ejecuta en el servidor EIS.

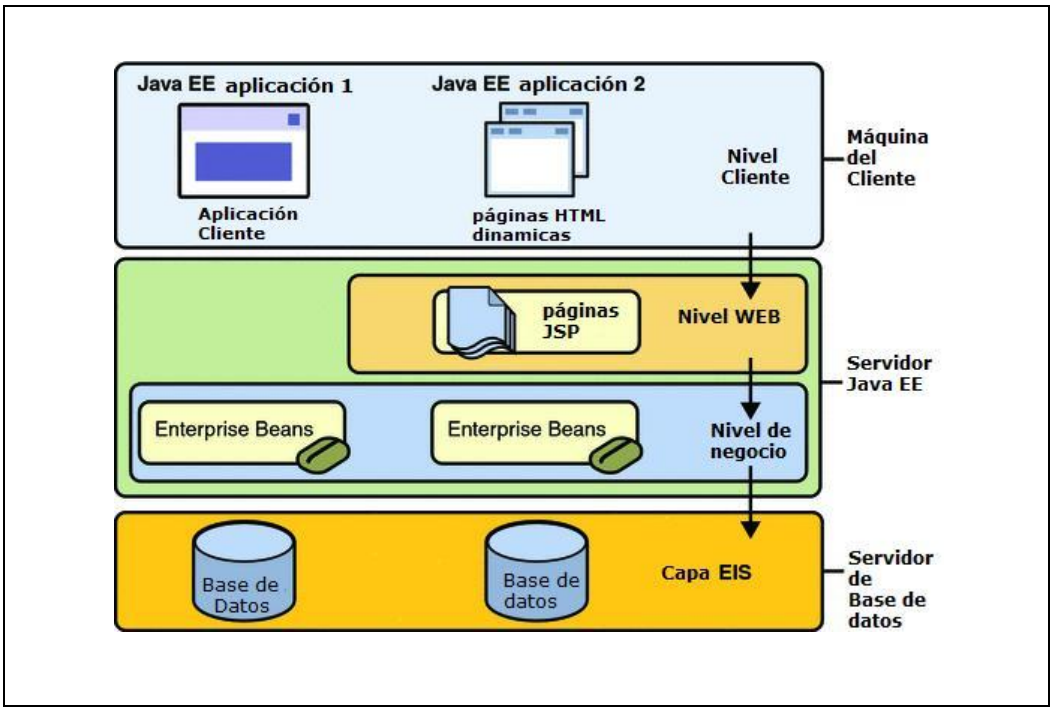


Figura (2.2.3) muestra dos aplicaciones distribuidas Java EE en las que se aprecia la división de niveles de los componentes de JavaEE.

Aunque una aplicación Java EE puede consistir de los tres o cuatro niveles que se muestran en la Figura (2.2.3), generalmente se consideran tres niveles, ya que las aplicaciones están distribuidas en tres lugares: las máquinas cliente, el equipo del servidor Java EE, y las máquinas de la base de datos (maquinas de almacenamiento final).

Las aplicaciones desarrolladas en Java EE están basadas en el patrón de diseño MVC (Model-View-Controller), que se puede traducir como Modelo-Vista-Controlador.

El modelo.- Expone la funcionalidad de la aplicación, encapsula el estado de la aplicación, responde a consultas de estado y notifica a la vista cuando hay cambios de estado del sistema.

La vista.- La vista se encarga de desplegar lo que el modelo desea mostrar y puede solicitar al modelo actualizaciones permitiendo al controlador elegir la vista.

El controlador.- Se encarga de encapsular el flujo y comportamiento de la aplicación, mapea las acciones para los usuarios del modelo, responde a las consultas del estado de la aplicación.

2.2.3.2 Contenedores Java EE

La arquitectura Java EE independiente de plataforma y basada en componentes hace sencillo construir aplicaciones empresariales ya que la lógica de negocio es organizada en componentes reutilizables. Además, java EE provee servicios subyacentes en forma de un contenedor para cada tipo de componente, que hace que los esfuerzos de programación se concentren en la lógica de negocio y no en programar estos servicios.

Definición: Los contenedores son interfaces entre un componente y una funcionalidad de plataforma de bajo nivel la cual soporta al componente, el proceso de ensamblado se especifican propiedades de cada componente, estas propiedades configuran servicios con seguridad, administración de transacciones, y conectividad remota. (Java EE, 2010)

Tipos de contenedores	
Servidor Java EE	Parte de un producto Java EE cuando se está ejecutando. El servidor Java EE provee los servidores de EJB y web.
Contenedor **Enterprise JavaBeans (EJB).**	Administra la ejecución de Enteprise para las aplicaciones Java EE. El servidor Java EE provee los servidores de EJB
Contenedor Web	Administra la ejecución las páginas JSP y los componentes servlets. Los componentes Web y sus contenedores se ejecutan en el servidor Java EE.
Contenedor de aplicación cliente	Administra la ejecución de los componentes de aplicación del cliente. La Aplicación cliente y sus contenedores se ejecutan en el cliente.
contenedor Applet	Administra la ejecución de applets. Consiste en un navegador web y Java Plug-in que se ejecutan juntos en el cliente.

Tabla (2.2.5) Tipos de contenedores en JavaEE.

2.2.3.3 Servlet

Los componentes web de Java EE son los servlets y las páginas creadas con la tecnología JSP o Java Server Pages, los servlets son clases Java que dinámicamente procesan las peticiones web y construyen las respuestas, éstos realizan tareas similares a los ***CGI*** pero utilizan un ambiente diferente.

Los servlets se encargan principalmente de procesar el ***HTTP request*** y generar la ***HTTP response*** dinámicamente, los servlets ofrecen rendimiento y escalabilidad ya que la plataforma en la que se desarrollan es Java, sin embrago se debe tener cuidado al manejar los servlets, ya que no se debe abusar de la inclusión de lógica de negocios o presentación en estos.

2.2.3.4 Java Server Pages (JSP)

Las páginas JSP son documentos basados en texto que se ejecutan como servlets a la hora que se carga dicha página permitiendo un acercamiento natural al contenido HTML estático, básicamente las páginas JSP contienen código Java embebido y HTML, el código Java es el encargado de procesar la respuesta HTTP.

En la tecnología JavaServer Pages, las acciones son elementos que pueden crear y acceder a objetos del lenguaje de programación y afectan a la secuencia de salida. La especificación JSP define 6 acciones estándar que deben ser realizadas por cualquier aplicación compatible con JSP.

2.2.3.5 Bibliotecas de Etiquetas (Tag Libs)

Además de las acciones estándar, la tecnología JSP apoya el desarrollo de módulos reutilizables llamados: acciones personalizadas (*custom actions*). Una acción personalizada se invoca mediante el uso de una etiqueta personalizada dentro de una página JSP. Una biblioteca de etiquetas **TagLib** es por tanto una colección de etiquetas personalizadas.

Algunos ejemplos de tareas que pueden realizar acciones personalizadas incluyen procesamiento de formularios, acceso a bases de datos y otros servicios de la empresa como el correo electrónico, directorios y control de flujo.

Algunas características de las etiquetas personalizadas son:

- Es posible individualizar las etiquetas a través de los atributos pasados desde la página que llama.

- Tienen acceso a todos los objetos disponibles en las páginas JSP.
- Pueden modificar la respuesta generada por la página que lo llama.
- Pueden comunicar entre sí a través de un componente JavaBeans.
- Pueden ser anidadas una dentro de otra, teniendo en cuenta las complejas interacciones dentro de una página JSP. (Java EE,2010)

2.2.4 Apache Struts

2.2.4.1 Introducción

Apache Struts es un ***framework*** para aplicaciones web de código abierto desarrollado con Java EE que utiliza y extiende el API de Java servlet para el desarrollo de sistemas web sobre la arquitectura de modelo-vista-controlador (MVC).

El objetivo de Struts es separar de manera transparente el *modelo o lógica de negocios* (la lógica de la aplicación que interactúa con las bases de datos) de la *vista* (páginas HTML que son presentadas al cliente) y del *controlador* (instancia que maneja la información entre la vista y el modelo).

2.2.4.2 El modelo

El modelo o lógica de negocio puede a su vez ser dividido en dos subsistemas: el estado interno del sistema y las acciones que pueden ser tomadas para cambiar el estado.

El estado interno del sistema es representado por un conjunto de uno o más JEBeans los cuales pueden ser reutilizados cada vez que se necesite en la aplicación, la información que encapsula los JEBeans regularmente es modificada siguiendo las reglas o acciones de negocio.

2.2.4.3 La vista

Dentro de Struts las vistas se desarrollan principalmente a través de la tecnología JSP, las vistas se crean a partir de “plantillas JSP”, dichas plantillas contienen xHTML estático y la capacidad de insertar contenido de forma dinámica basados en la interpretación de etiquetas especiales: las librerías de etiquetas o ***Tag Libs***.

El ***framework*** Struts incluye un conjunto Tag Libs que facilitan el desarrollo de interfaz de usuario e interactúan directamente con los actions Bean Forms y los actions Bean Validator Forms, esta interacción provee funcionalidades como la validación y la captura automática de datos.

La vista por tanto es separada de la lógica de negocios a través de las etiquetas (lo cual evita que se mezcle código java en las JSP) y de los archivos de recursos de la aplicación indicados en el struts-config.xml.

2.2.4.4 El controlador

Una parte del controlador es provista por la arquitectura Struts, dicho *controlador* (un tipo especial de servlet conocido como ***ActionServlet***), está enfocado a recibir la petición del cliente (regularmente un usuario utilizando un navegador web), decide qué función de la lógica de negocios es ejecutada y

delega la responsabilidad para producir la siguiente vista.

El actionServlet debe ser configurado para definir el conjunto de posibles rutas a las que responderá dicho controlador, cada ruta es conocida como ActionMapping. Un ActionMapping además de definir la ruta, define el nombre de la clase Action con la que está directamente relacionada.

Las clases Action son subclases de org.apache.struts.Action y éstas se encargan de:

- Encapsular las llamadas a la lógica de negocios.
- Interpretar el resultado de la petición.
- Despachar el control al componente adecuado que creará la respuesta.

2.2.4.5 Complementos de Struts (Struts Plug-Ins)

Se conocen como Plug-Ins a las interfaces que extienden la clase Action y que fácilmente pueden agregarse dentro del ciclo de vida de un ActionServlet. Esta interfaz define dos métodos :init() y destroy(), cuando la aplicación es levantada se llama el método init() y cuando es apagada se llama el método destroy().Un uso común que se le da a los plug-ins es para configurar o cargar datos a la aplicación cuando dicha aplicación esta arrancando.

En tiempo de ejecución, cualquier configuración de recursos realizada por la función init() puede ser accesible por las clases Action o clases de lógica de negocios. La interfaz de plug-ins permite a los recursos ser configurados, pero no proporciona ninguna forma especial para acceder a ellos. En la mayoría de los casos, el recurso se almacena en el contexto de aplicación, en virtud de una clave conocida, donde otros componentes pueden encontrarlo.