ESTADO del ARTE FRAMEWORK JAVA a PARTIR de PRÁCTICAS ACADÉMICAS.
STATE ART FRAMEWORK JAVA From ACADEMIC PRACTICES.

Mario Dustano Contreras Castro[1]

Resumen - Este artículo evidencia la vivencia hacia la exploración, profundización, fracasos, seguimientos, retroalimentación de los Fundamentos Teóricos, Conceptuales, Epistemológicos de un entorno o ambiente de trabajo para el desarrollo de software, como lo es Framework, hacia el interior de espacios académicos en modalidad distancia como presencial.
Se contextualiza inicialmente en Frameworks hacia el interior de dos programas: Ingeniería en Informática (Modalidad Distancia) en la Universidad Santo Tomás e Ingeniería de Sistemas (Modalidad Presencial) en la Universidad Autónoma de Colombia. Luego, se colocan unos referentes que fueron prácticas o talleres orientados a la exploración, la experiencia y el aprendizaje en Frameworks Java.

Palabras claves: Exploración, Framework, Modalidad Distancia , Modalidad Presencial, Patrones de Diseño , Prácticas.

Abstract - This paper brings the experience to the exploration, deepening, failures, monitoring, feedback Theoretical Foundations, Conceptual, Epistemological an environment or working environment for software development, as is Framework, into academic spaces in distance mode and face. It was initially contextualized in Frameworks into two programs: Computer Engineering (Distance Mode) at the University St. Thomas and Systems Engineering (Modality) at the Autonomous University of Colombia. Then, a few references that were practical or workshops to explore, experience and learning in Java Frameworks are placed.
Keywords: Exploration, Framework, Distance Modality, Modality, Design Patterns, Practices.

1. INTRODUCCION

Según [1] los patrones de diseño en la construcción de software se especifican acorde a la etapa donde se aplican: modelos de dominio, patrones arquitecturales (estilos), patrones de diseño, patrones de código específicos a una tecnología (idioms), patrones de diseño implementados en una API (Frameworks), patrones de procesos.

Antes de formular que es Framework, se hace necesario definir que la Arquitectura de Componentes Software (Fig. 1) es el conjunto de componentes[2] y sus relaciones[3], sus interfaces[4], obligaciones, servicios o contratos[5].

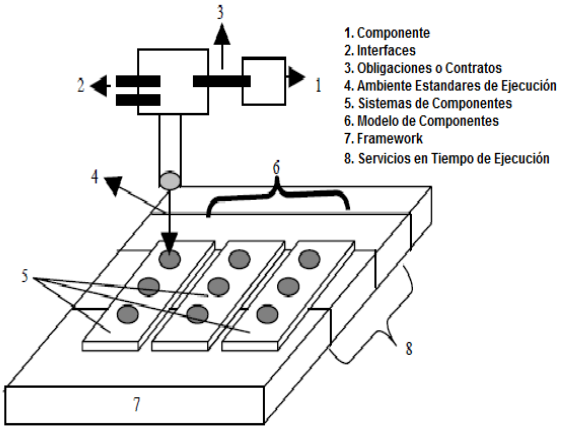

Fig. 1. Arquitectura de Componentes Software. Adaptación [1]

Un Framework se define como un cascaron o una estructura en bruto donde se van colocando los componentes acorde con una necesidad, requerimiento o una exigencia basado en un análisis de alto nivel, como también, en una valoración crítica. Entones, un Framework es una abstracción de un componente de software (su construcción se basa en la experiencia) para resolver un problema en UN CONTEXTO (ojo no confundir con PATRON DE DISEÑO que es para resolver un problema en CUALQUIER contexto).

En Modalidad Presencial y Modalidad Distancia se realiza la aplicación, la experimentación y evaluación a partir de programas desarrollados y codificados en Lenguaje Java (Framework Java).

[1] Ingeniero de Sistemas. Especialista en Multimedia Educativa. Maestría en Edumatica. Docente de Tiempo Completo, Programa Ingeniería en Informática, Facultad de Ciencia y Tecnologías, VUAD, Universidad Santo Tomas Sede Bogotá, mariocontreras@ustadistancia.edu.co. Docente cátedra programa Ingeniería de Sistemas, Universidad Autónoma de Colombia, mario.contreras@fuac.edu.co

[2] Representa una unidad coherente de funcionalidad

[3] Indican que aspectos de un componente son usados por otros componentes y cómo la comunicación entre componentes procede sobre el tiempo

[4] Es una estructura que especifica obligaciones o contratos de un Componente Software

[5] Conjunto o una colección de operaciones a realizar por un componente software

Fig. 2. Aula Virtual Framework (Modalidad Distancia). Fuente Propia

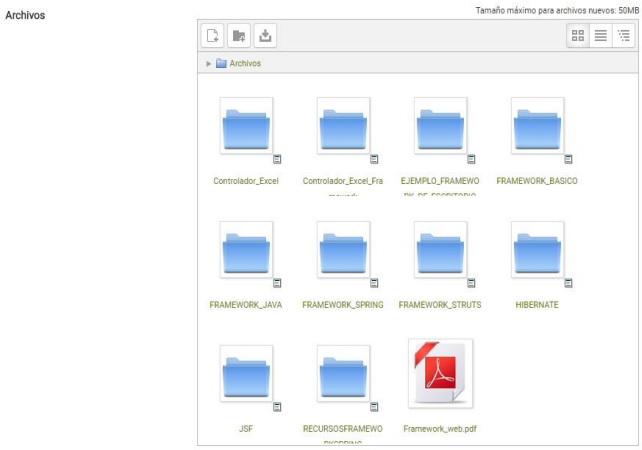

Fig. 3. Archivos Tema Framework (Modalidad Presencial). Fuente Propia

2. DESARROLLO DEL ESTADO DEL ARTE

A. Primer Referente. Excel API

1) *Teoría Comprobada*. Excel Api[6] es un componente que maneja el acceso y edición de un archivo Excel en JAVA

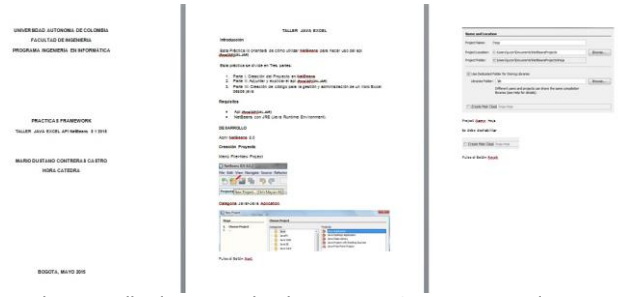

Fig. 4. Taller java Excel api NetBeans 8. Fuente Propia

2) *Objetivos del Taller.* Primer Objetivo es búsqueda por internet y adjuntar el controlador Excel Api (Controlador Jxl.Jar) a un Proyecto Java NetBeans. Segundo Objetivo es el desarrollo de una clase que administre la Hoja Excel (Hojas). Tercer Objetivo es la creación de un archivo Excel.

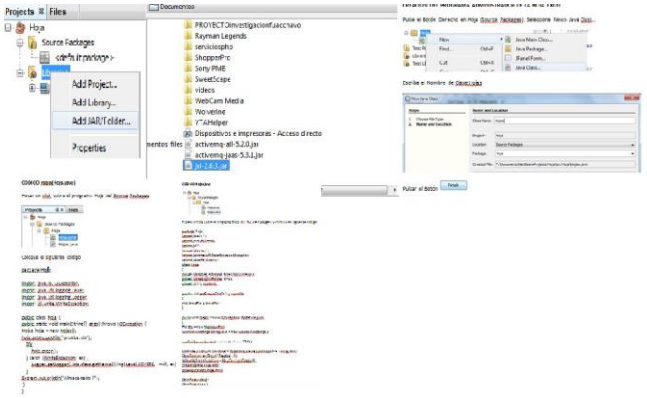

Fig. 5. Pantallas Taller Excel API. Fuente Propia

B. Segundo Referente. Interface MySql

1) *Teoría Comprobada.* Conjunto de API (JDBC[7] y JNDI[8]) para la conexión a Bases de Datos Relacionales con programas Java.

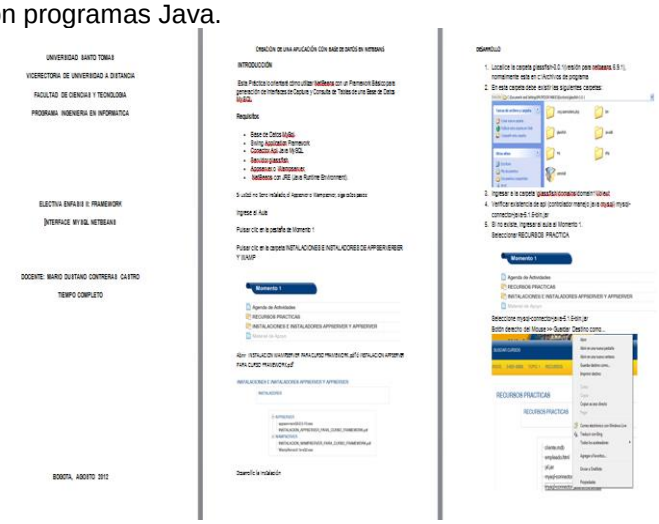

Fig. 6. Práctica Interface MySQL NetBeans. Fuente Propia

2) *Objetivo de las Prácticas.* Realizar en programas Java las operaciones de Consulta, Edición, Borrado y Adición (CRUD) sobre una Tabla de una Base de Dato Relacional MySQL.

Se tuvo en cuenta para las prácticas los siguientes ítems:

1. Conexión a una Base de Datos a partir de la API JDBC. Entonces, la aplicación Java obtiene una conexión a la Base de Datos para realizar operaciones de Consulta, Edición, Borrado y Adición (CRUD), así mismo, el Servidor de aplicaciones obtiene una referencia de una conexión física a la Base de Datos.
2. Uso del API JNDI para facilitar el nombre del directorio donde está localizado el recurso JDBC.

[6] Conjunto de subrutinas, funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción

[7] API [2] que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java, independientemente del sistema operativo donde se ejecute o de la base de datos a la cual se accede, utilizando el dialecto SQL del modelo de base de datos que se utilice

[8] Java Naming and Directory Interface[2] es una Interfaz de Programación de Aplicaciones (API) para servicios de directorio

Actividades Desarrolladas.

- Actualizar componentes GlassFish
- Modelo de Bases de Datos(Conceptualización)
- Servidor GlassFish
  Consola de Administración de Glass Fish
  Activar Servidor GlassFish desde NetBeans
- Registro MySQL Server
- Conexión a la Base de Datos MySQL
- Creación de la Aplicación de Bases de Datos en NetBeans
- Ejecución

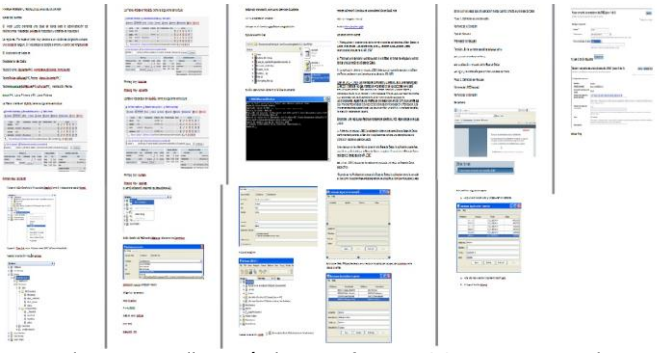

Fig. 7. Pantallas Práctica Interface MySQL. Fuente Propia

C. Tercer Referente. Hibernate Java

1) Teoría Comprobada. Persistir objetos Java es la manera de almacenar permanentemente objetos en una tabla de una base de datos relacional o sea convertir objetos Java a columnas/registros de una base de datos y viceversa. Por lo tanto, la persistencia es serializar objetos Java[9] estructurados en forma de árbol a una base de ratos relacional estructurada de forma tabular y viceversa requiriendo el mapeo de los objetos Java a columnas y registros de la base de datos de una manera optimizada en velocidad y eficiencia.

Hibernate Java es una herramienta de persistencia "objeto-java-a-base-de-datos" o de Mapeo Objeto-Relacional (ORM) haciendo de archivos declarativos (XML) para representar la transformación de los datos de un modelo relacional a un modelo orientado a objetos (Pojo) y viceversa.

[9] Consiste en obtener una secuencia de bytes que represente el estado de dicho objeto

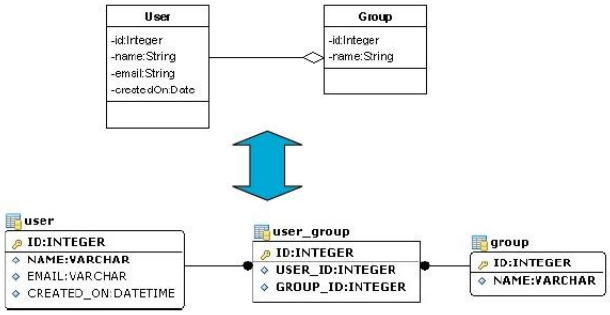

Fig. 8. Hibernate Java. Fuente Propia

En la Figura 8 se persisten las Tablas User y Group para generar los objetos user, user_group, group.

Fig. 9. Generación Archivo de Mapeo Relacional. Fuente Propia

2) Objetivo de las Prácticas. Utilizar Framework Hibernate en NetBeans para generación de Interfaces de Captura y Consulta de Tablas de una Base de Datos MySQL.

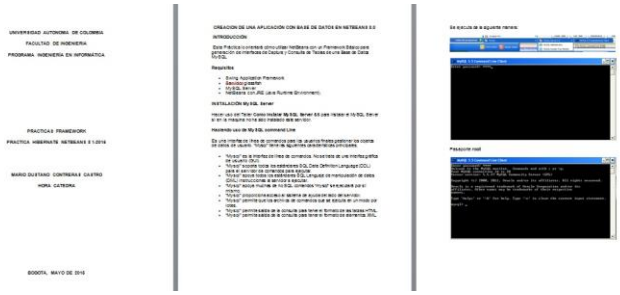

Fig. 10. Práctica FrameWork Hibernate NetBeans 8.0 . Fuente propia

Actividades Desarrolladas.

- Instalación MySQL server
- Conceptualización.
  Que es Hibernate
  El Lenguaje de Consulta Hibernate
  Documentación Hibernate
- Descarga MySQL Connector
- Registro del Servidor MySQL en NetBeans
- Conexión a la Base de Datos MySQL en NetBeans
- Creación de la aplicación Hibernate en NetBeans
  Primer Paso. Creación de la entidad Cliente
  Segundo Paso. Escritura del Archivos de Mapeo y Configuración
  Tercer paso. Creación Archivo de Mapeo Cliente.hbm.xml

Cuarto Paso. Escritura del Archivo Configuración Hibernate
Quinto Paso. Adicionar librería Hibernate JPA(Java Persistence Applications) como el Controlador mySql Conector
Sexto Paso. Crear archivo de sesión y de ayudante (helper) hibernate
Séptimo Paso. Creación de un Archivo Menu
Octavo Paso. Actualizar la clase main

- Ejecución

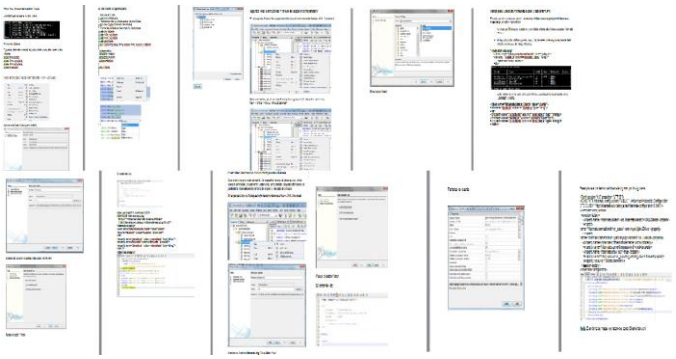

Fig. 11. Pantallas de Practica_Hibernate_NetBeans_8. Fuente Propia

D. Cuarto Referente. Frameworks Struts

1) Teoría Comprobada. Framework Struts es un framework que implementa el patrón de arquitectura MVC en Java. El patrón de arquitectura MVC (Model-View-Controller) se define en el Model (Objetos de Negocio), la View (interfaz con el usuario u otro sistema) y el Controller (controlador del workflow[10] de la aplicación).

El Framework Struts 1.x implementa el patrón de arquitectura MVC separando el modelo o workflow de la aplicación (se puede programar desde un archivo XML), del modelo de objetos de negocio y de la generación de interfaz.

Las acciones se ejecutan sobre el Model (Objetos de Negocio) y se implementan basándose en clases predefinidas por el framework a partir de un patrón de diseño Facade. La generación de la View (interfaz con el usuario u otro sistema) se realiza a partir de un conjunto de Tags predefinidos por Struts generando ventajas de mantenimiento y de desempeño (pooling de Tags, caching, etc). Así mismo, separa el desarrollo de interfaz del workflow y lógica de negocio permitiendo desarrollar ambas en paralelo.

El Framework Struts 1.x permite la reutilización de componentes y puede hacer uso de múltiples interfaces de usuario (Html, sHtml, Wml, Desktop applications, etc.) y de múltiples idiomas, localismos, etc.

El Controller (controlador) Struts se encarga de tres tareas:

1. Validaciones simples: extendiendo la clase ActionForm, se realiza validaciones simples sin acceder al modelo, haciendo uso de expresiones regulares (comprobación formato y longitud de datos)
2. Validaciones Complejas: extendiendo la clase Action, se comprueba las reglas de negocio (modelo). Por ejemplo: Se realizan consultas contra la base de datos y se obtienen los errores de integridad, etc.

Control de flujo o de Navegación: a través de un archivo de configuración (struts-conf) se gestiona el flujo de navegación entre páginas, que también se logra extendiendo la clase ActionForm y Action.

Fig. 12. Práctica Struts 1.x. Fuente Propia

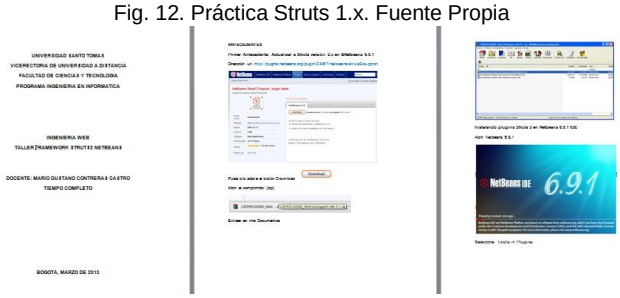

Fig. 13. Práctica Struts 2.x. Fuente Propia

El Framework Struts 2.x establece:

1. Cuáles deben ser las peticiones o solitudes a recibir desde el usuario.
2. El elemento FilterDispatcher(administrador de responsabilidades) determina cual es el action(la clase base de struts 2.x) que deberá responder a una determinada petición o solicitud. El framework struts posee la estructura para que el administrador de responsabilidades (FilterDispatcher) determine qué action es el responsable de recibir la petición, procesarla y publicar el resultado de su ejecución.

En la Figura 14 se observa una clase Action:

- Es una clase java
- Esta clase java cumple con normatividad de variables javabeans: nombre en minúscula, acceso de variables private, métodos getter y setter
- El Método execute() retorna ERROR cuando no cumple con lo solicitado. Retorna SUCCESS cuando cumple con lo solicitado.

[10] Estudio de los aspectos operacionales de una actividad de trabajo: cómo se estructuran las tareas, cómo se realizan, cuál es su orden correlativo, cómo se sincronizan, cómo fluye la información que soporta las tareas y cómo se le hace seguimiento al cumplimiento de las tareas

```
public class AnnotationAction extends ActionSupport {

      private String username = null;
      private String password = null;


      @RequiredStringValidator(message="Supply name")
      public String getUsername() {
            return username;
      }


      public void setUsername(String value) {
            username = value;
      }


      @RequiredStringValidator(message="Supply password")
      public String getPassword() {
            return password;
      }

      public void setPassword(String value) {
            password = value;
      }

      public String execute() throws Exception {

        if(!getUsername().equals("Admin") || !getPassword().equals("Admin")){
           addActionError("Invalid user name or password! Please try again!");
           return ERROR;
        } else{
          return SUCCESS;
        }
      }
}
```

Fig. 14. Ejemplo de una clase Action. Fuente Propia

Acorde con la Tabla 1 (Tabla de Responsabilidades), se debe declarar: el nombre de la responsabilidad (Action), el nombre de la clase Action, el nombre del interceptor[11], el nombre de la página JSP que recibe la petición, el nombre de la página JSP donde se publican los errores, el nombre de la página JSP donde se publica el resultado de la ejecución con éxito, como también, en algunos casos el llamado a un JavaBean para realizar un modelo de datos o lógica del negocio.

TABLA 1.
TABLA DE RESPONSABILIDADES

Responsabilidad (Action)		Interceptor	Nombre de Página JSP Resultado			Nombre del JavaBean
Nombre	Clase		Petición	Error	Éxito	
Crear	AccionCrear		Entrada.jsp	Entrada.jsp	Creacion.jsp	Cuenta
Actualizar	AccionActualiza		Transaccion.jsp	Transaccion.jsp	Actualiza.jsp	Cuenta
Menu			index.jsp			

3. Cada solicitud o petición dentro del servidor se relaciona con un Action (por nombre de responsabilidad). Entonces, una vez recibida una solicitud o petición se ejecuta el interceptor siempre y cuando el Action tenga asociado uno. Luego, se ejecuta la clase action haciendo uso del método execute() y retornando un resultado. De acuerdo a este resultado (error o éxito en dar respuesta a la petición o solicitud) se determina la página a publicar (Dispatcher).
4. Se publica el resultado en el cliente.

Un ejemplo, de acuerdo a la lógica a emplear en el FilterDispatcher(administrador de responsabilidades) se describe a partir de la siguiente tabla:

[11] Clases que se emplean para especificar el ciclo de vida de una petición, y están pensadas para añadir funcionalidad extra a las acciones. Los interceptores se pueden configurar para que ejecuten diferentes servicios, a saber, workflows, validaciones, upload de archivos, etc. Entonces, los interceptores en struts2 son clases que se ejecutan antes o después que se invoque un Action

TABLA 2.
TABLA EJEMPLO DE RESPONSABILIDADES

Responsabilidad (Action)		Interceptor	Nombre de Página JSP Resultado			Nombre del JavaBean
Nombre	Clase		Petición	Error	Éxito	
Crear	AccionCrear	Nuevo	Entrada.jsp	Entrada.jsp	Creacion.jsp	Cuenta

En el archivo struts.xml, se debe escribir el tag <action>…</action> para declarar los Actions de la siguiente manera:

```
<action name="Crear">
   <result>/Entrada.jsp</result>
</action>
<action name="Crear" class="Action.AccionCrear">
   <result name="input">/Entrada.jsp</result>
   <result name="error">/Entrada.jsp</result>
   <result>/Creacion.jsp</result>

</action>
```

También, en el archivo struts.xml, se debe escribir el tag <interceptors>…</interceptors> para declarar los interceptores de la siguiente manera:

```
<interceptors>
<interceptor name="Nuevo" class=" Action.Nuevo "/>
</interceptors>
```

Se puede decir entonces:

- El nombre de la responsabilidad o action se llama Crear
- La clase action se denomina AccionCrear
- La clase interceptor es Nuevo.class
- entrada.jsp es la página JSP donde se recibe la petición dentro del servidor . También, es donde se retorna el control si existe error en dar respuesta a la petición o solicitud
- creación.jsp es la página JSP donde retorna el control si existe éxito en dar respuesta a la petición o solicitud
- Cuenta se encuentra en el paquete Beans. Se hace uso de un javabean Cuenta si existe éxito en responder la petición o solicitud.

El llamado del action Crear se hace desde una página JSP de la siguiente manera:

```
<s:a href="Crear.action">Creacion Cuenta</s:a>
```

El action Crear es el responsable de recibir la petición, procesarla y publicar el resultado de su ejecución de acuerdo con la siguiente lógica:

- Se realiza el interceptor Nuevo.class
- Se ejecuta entrada.jsp. El usuario llenara los campos solicitados.
- Cuando se pulsa el botón submit, se hace un llamado a la clase AccionCrear
- En AccionCrear se actualizan las variables de clase mediante los métodos setter y después se realiza el método execute().
- si existe error en dar respuesta a la petición o solicitud se despacha o publica la página JSP entrada.jsp
- si existe éxito en dar respuesta a la petición o solicitud se despacha o publica la página JSP creacion.jsp

Fig. 15. Práctica_Framework Struts2 NetBeans 8. Fuente Propia

El Controller (controlador) Struts 2.x se encarga de realizar tres tareas similares a Strut 1.x:

1. Validaciones simples.
2. Validaciones Complejas.
3. Control de flujo o de Navegación.

2) *Objetivo de las Prácticas.* Comprobar el patrón de arquitectura MVC (Model-View-Controller) hacia el interior del Framework Strut en NetBeans.

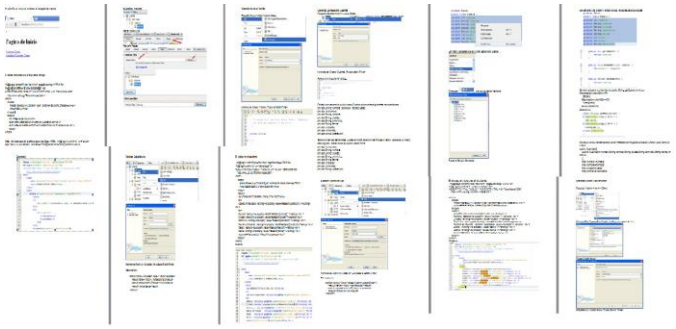

Fig. 16. Pantallas de Práctica_Framework Struts2 NetBeans 8. Fuente Propia

Actividades Desarrolladas.

- Actualización de Strut 2.x
- Conceptualización.
  Arquitectura Strut 2.x
  Struts 1 vs Struts 2
  Patrón de Diseño Cadena de Responsabilidades Descarga MySQL Connector
- Framework Strut en Netbeans
  Primer Paso. Creación de Proyecto
  Segundo Paso. Editando el archivo struts.xml
  Tercer Paso. Eliminar index.html
  Cuarto Paso. Adicionar index.jsp
  Quinto Paso. Actualizar web.xml
  Sexto Paso. Creando la clase Validar
  Séptimo Paso. Creando JavaBeans Cuenta
  Octavo Paso. Creando Páginas JSP Petición y Vista
  Noveno Paso. Edición Transaccion.jsp
  Decimo Paso. Edición Actualiza.jsp
  Décimo Primer Paso. Edición Creacion.jsp
  Décimo Segundo Paso. Creando Action AccionCrear
  Décimo Tercer Paso. Modificar el método execute()
  Décimo Cuarto Paso. Creando Action AccionActualiza
  Décimo Quinto Paso. Modificar el método execute()
- Ejecución

E. Quinto Referente. Frameworks Spring

1) Teoría Comprobada. Spring es un framework liviano que generalmente los objetos que se programan no tienen dependencias en clases específicas de Spring. Sus características principales son inyección de dependencias **y** programación orientada a aspectos.

Inyección de dependencias. El objetivo es lograr un bajo acoplamiento entre los objetos de una aplicación. Con **el Patrón Inversión de Control[12] (IoC)**, los objetos no crean o buscan sus dependencias (objetos con los cuales colabora) sino que éstas son dadas al objeto. El contenedor (la entidad que coordina cada objeto en el sistema) es el encargado de realizar este trabajo al momento de crear el objeto. Se invierte la responsabilidad en cuanto a la manera en que un objeto obtiene la referencia a otro objeto.

De esta manera, los objetos conocen sus dependencias por su interfaz. Así la dependencia puede ser intercambiada por distintas implementaciones a través del contenedor. En resumen, se programa orientado a interface y se inyecta las implementaciones a través del contenedor.

Programación orientada a aspectos (AOP). Se trata de un paradigma de programación (Codigo, 2012) que intenta separar las funcionalidades secundarias[13] de la lógica de negocios.

Módulos de Spring. En la figura 17 se muestran los módulos del Spring. En su núcleo (Core) se encuentra el BeanFactory – el contenedor fundamental de Spring y quien se encarga de la inyección de dependencias. El contenedor ApplicationContext se basa en BeanFactory,

[12] IoC permite instanciar una clase (crear objetos) olvidando las dependencias que tiene. Para eso, primero se registran las clases en un Contenedor (IoC Container). Segundo, haciendo uso de Dependency Injection, se generan los Constructores de las clases registradas en el Contenedor (IoC Container), para así, recuperar fácilmente cualquier clase en cualquier otra clase. IoC también es famoso por el principio de HollyWood: “don’t call us, we’ll call you” (No nos llames, nosotros le llamaremos.)

[13] “cross-cutting concerns” algo que se traduciría como “preocupaciones transversales”

eventos de ciclo de vida, validación y mejor integración con AOP.

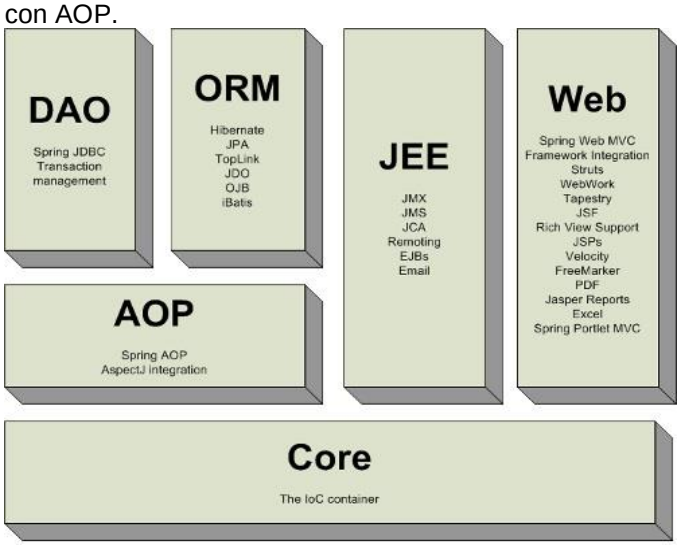

Fig. 17. Modulo del Framework Spring. Fuente http://www.j2eebrain.com/wp-content/uploads/Spring-Framework.png

AOP – provee la implementación de AOP, permitiéndonos desarrollar interceptores de método y puntos de corte para desacoplar el código de las funcionalidades transversales.
DAO - Provee una capa de abstracción sobre JDBC, abstrae el código de acceso a datos de una manera simple y limpia. Tiene una capa de expeciones sobre los mensajes de error provistos por cada servidor específico de base de datos. Además cuenta con manejo de transacciones a través de AOP.
ORM – Provee la integración para las distintas APIs de mapeo objeto-relacional incluyendo JPA, JDO, Hibernate e iBatis.
JEE – Provee integración con aplicaciones Java Enterprise Edition así como servicios JMX, JMS, EJB, etc.
Web – Módulo que aporta clases especiales orientadas al desarrollo web e integración con tecnologías como Struts y JSF. Cuenta con el paquete Spring **MVC,** una implementación del conocido patrón de diseño aplicando los principios de Spring.

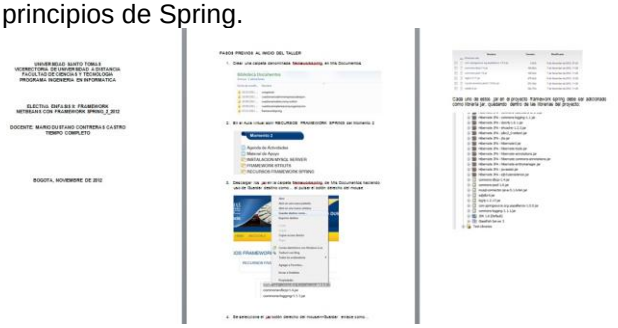

Fig. 18 Práctica Framework Spring Netbeans 6.9. Fuente Propia.

2) *Objetivo de las Prácticas.* Desarrollar los módulos del Framework Spring en NetBeans.

Actividades Desarrolladas.

- Conceptualización
  Patrón Inversión de Control (IoC)
  Que es Hibernate
  Que es Framework Spring
- Instalación y registro MySQL server
- Desarrollo de Framework Spring en NetBeans
  Primer Paso Creación de JavaBean Cliente
  Segundo Paso Creación de Interface InterfaceCliente
  Tercer Paso. Escritura del Archivo de Mapeo
  Cuarto Paso. Crear archivo Hibernate
  Quinto Paso. Crear Archivo Interceptor
  Sexto Paso. Crear Archivo Test
  Séptimo Paso. Creación de JSP Actualiza
  Octavo Paso. Creación de JSP Ayudante
  Noveno Paso. Modificar redirect.jsp
- Ejecución

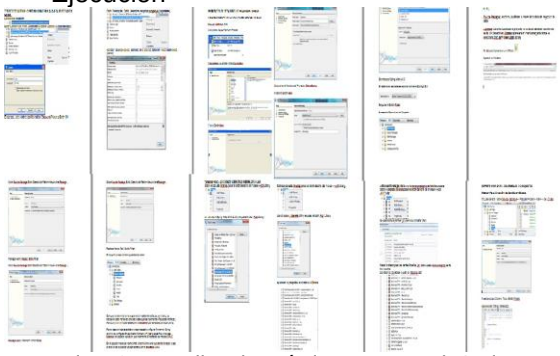

Fig. 19. Pantallas de Práctica Framework Spring Netbeans 6.9. Fuente Propia

F. Sexto Referente. Framework para la persistencia de datos

1) Teoría Comprobada. JPA[14] establece una interface común que es implementada por el proveedor de persistencia eclipse.

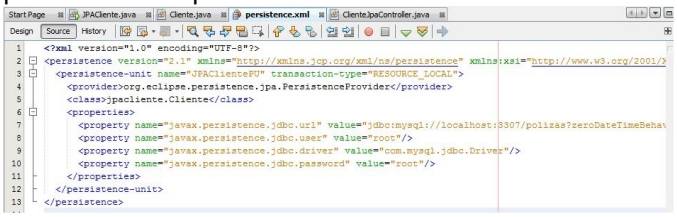

Fig. 20. Ejemplo Archivo de configuración persistence.xml. Fuente Propia.

En el archivo de configuración persistence.xml (Figura 20) se define:

• Persistence-unit, atributo name: define una unidad de persistencia, es obligatorio darle un nombre, para poder crear un EntityManager. El EntityManager es el encargado de manejar la persistencia de los datos.

• Provider: si se desea usar Hibernate o no como implementación JPA. En esta práctica se selecciona el proveedor eclipse

• class: debe existir una etiqueta class por cada clase que se desee persistir, es decir, una por cada entidad se formará una unidad de persistencia.

[14] API de persistencia desarrollada para la plataforma Java EE. Es un Framework del lenguaje de programación Java que maneja datos relacionales en aplicaciones usando la Plataforma Java en sus ediciones Standard (Java SE) y Enterprise (Java EE).

• Dentro de las etiquetas properties, se define propiedades prioritarias de la implementación JPA. Entre ellas se destacan:

1. hibernate.connection.url: define la URL de conexión a la base de datos.
2. User
3. Password
4. Driver

2) Objetivo de las Prácticas. Desarrollo de JPA a partir de una Base de Datos Relacional MySQL.

Fig. 21. Practica_JPA_NetBeans_8. Fuente Propia

Actividades Desarrolladas.

- Instalación MySQL server
- Creación de una Base de Datos con sus respectivas tablas
- Conceptualización
  Que es Hibernate
  Interfaz de Persistencia Java (JPA)
  Diferencia entre Hibernate y JPA
- Desarrollo en NetBeans
  Registro Servidor MySQL
  Conexión a la Base de Datos MySQL
  Creación de Aplicación JPA
- Ejecución

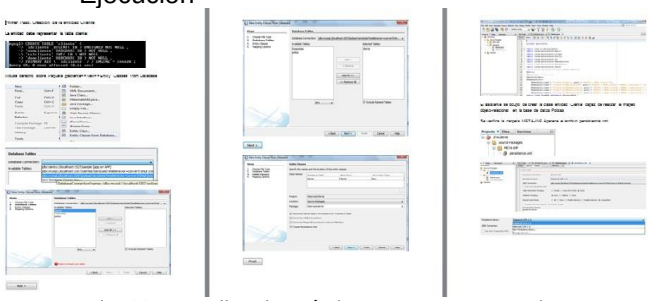

Fig. 22. Pantallas de Práctica JPA. Fuente Propia

G. Séptimo Referente. Practica JSF (Java Server Face)

1) Teoría Comprobada. JSF (Java Server Faces) es un Framework MVC para desenvolvimiento web en lenguaje de programación Java.

Principales características de JSF

• Es un Framework que fue definido por JCP (Java Community Process), que es la entidad que define las especificaciones de la evolución de la tecnología Java.

• Conjunto de componentes para la interfaz del usuario.

• Programadores pueden crear componentes adicionales.

• Componentes visuales se conectan con los datos del servidor

• Eventos de componentes visuales realizan funciones en el servidor. (ManagedBeans)

• Validadores y convertidores de datos, todos los datos que llegan al servidor vienen en formato texto, puede que nuestro usuario escriba una fecha de nacimiento, en JSF tenemos conversores automáticos.

• Soporte AJAX[15], puede mostrar información que proviene del servidor sin necesidad de recargar la página.

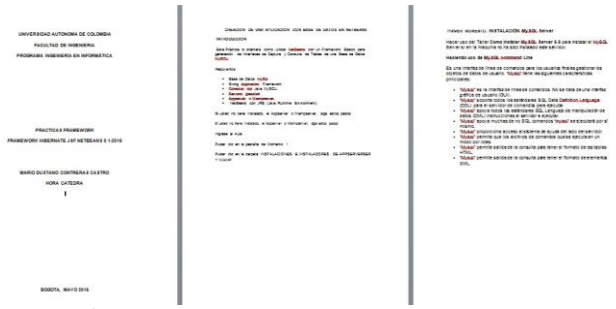

Fig. 23. Práctica FrameWork Hibernate JSF NetBeans 8. Fuente Propia

2) Objetivo de las Prácticas. Desarrollo de JPA Hibernate a partir de una Base de Datos Relacional MySQL.

Actividades Desarrolladas.

- Instalación MySQL server
- Creación de una Base de Datos con sus respectivas tablas
- Conceptualización
  Que es Framework Hibernate Java
  HQL: El lenguaje de consulta de Hibernate
- Desarrollo en NetBeans
  Registro Servidor MySQL
  Conexión a la Base de Datos MySQL
  Creación de Aplicación Hibernate
  Primer paso. Modificar el archivo de configuración de Hibernate
  Segundo paso. Crear archivo de sesión y de ayudante (Helper)
  Tercer paso. Generación de archivos de mapeo Hibernate y las clases Java
  Cuarto paso. Creación de los archivos Pojo y de Asignación
  Quinto paso. Edición de una clase ayudante (Helper)
  Sexto paso. Desarrollo de consultas y método de adición de registro
  Séptimo paso. Crear páginas web
  Octavo paso. Creación de archivos XML
- Ejecución

[15] Acrónimo de *Asynchronous Javascript and XML,* es decir, Javascript y XML Asíncrono.

Fig. 24. Pantallas Práctica FrameWork Hibernate JSF. Fuente Propia

G. Octavo Referente. Practica Jquery

1) Teoría Comprobada. jQuery es un Framework de Javascript, o ambiente de desarrollo de JavaScript que contiene librerías de funciones pre-desarrolladas (funciones programadas, probadas y que pueden ser utilizadas de manera más sencilla).

JavaScript haciendo uso del DOM[16] accede a una página web y sus objetos (párrafos, divisiones, tablas, formularios y campos), para así, consultar y modificar sus propiedades y métodos.

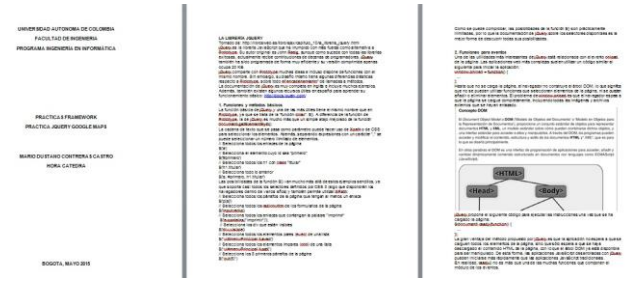

Fig. 25. Practica JQUERY Google Maps. Fuente Propia

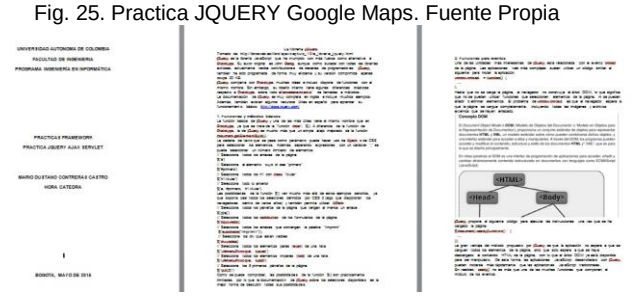

Fig. 26. Practica jQuery Ajax Servlet. Fuente Propia

2) Objetivo de las Prácticas. Desarrollo de un proyecto NetBeans con la inclusión de un llamado al Framework jQuery.

Actividades Desarrolladas

- Conceptualización
  Ajax
  jQuery
- Proyecto NetBeans
  Modificar index.HTML
  Colocar campos de Captura de Datos
  Búsqueda de bibliotecas de código abierto JavaScript
  Colocar en el tag <script>

Fig. 27. Tag para versión jquery . Fuente Propia

Adición de Hoja de Estilo (css)
Adición de archivos de datos
Creación de una clase Control
Creación del archivo servlet
Configuración archivo xml

- Ejecución

Fig. 28. Pantallas Practica JQUERY Google Maps. Fuente Propia

Fig. 29. Pantallas Practica jQuery Ajax Servlet. Fuente Propia

3. CONCLUSIONES

El aprendizaje exigido debe ser el mismo no importanto la modalidad Presencial o Distancia.

Los referentes bibliográficos de las prácticas exigen que el estudiante lea, consulte en la web y amplie la conceptualización.

El momento de la retroalimentación se diferencia en la modalidad. Cuando es modalidad presencial la retoralimentación es directa (2 sesiones semanales) y mediada por Aula Virtual. En modalidad distancia la retoralimentación directa se da a traves de 4 tutorias presenciales (cada una de 2 horas) y mediada por Aula Virtual, Foros, Correos Electronicos. Pero la oportunidad de ensayo y error en el desarrollo de prácticas es igual para ambas modalidades. Este ensayo y error ha permitido ampliar como profundizar,

[16] Document Object Model o DOM ('Modelo de Objetos del Documento' o 'Modelo en Objetos para la Representación de Documentos') es esencialmente una interfaz de plataforma que proporciona un conjunto estándar de objetos para representar, acceder y modificar el contenido, estructura y estilo de los documentos HTML y XML. El responsable del DOM es el World Wide Web Consortium (W3C).

relacionar los conceptos con la práctica, mejorar el orden de la práctica, como también, la exigencia de crear nuevas prácticas.
Entonces, los constructos conceptuales como de las practicas parten de una necesidad de aprendizaje y actualización permanente no importando la modalidad, sea, Presencial o Distancia.

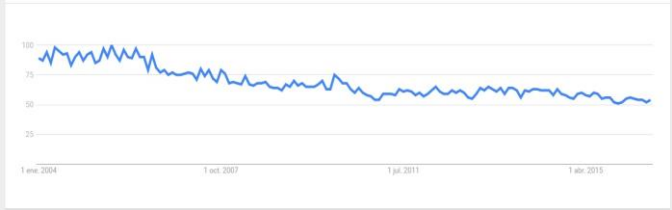

Fig. 30. Interés en Framework Java en el Mundo. Fuente https://www.google.es/trends

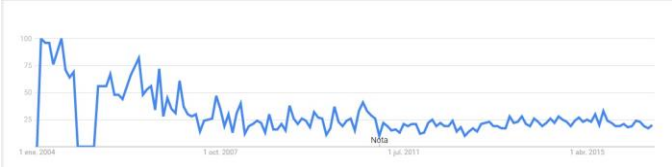

Fig. 31. Interés en Framework Java en Colombia. Fuente https://www.google.es/trends

En la Figura 31, se observa el interes hacia el desarrollo deFramework Java en los últimos cinco años(desde 2011) ha permanecido con algunas excepciones hasta un 25% de la población de programación java.

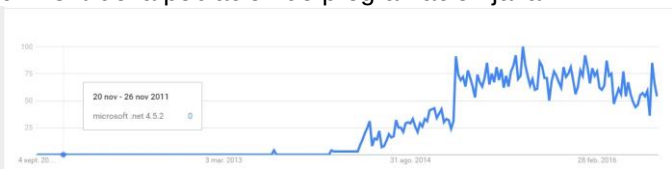

Fig. 32. Interés en .net Frameworks 4.5.2 en el Mundo. Fuente https://www.google.es/trends

Fig. 33. Interés en .net Frameworks 4.5.2 por Países. Fuente

https://www.google.es/trends

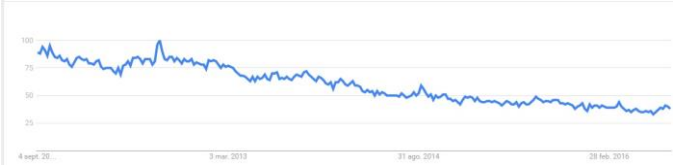

Fig. 34. Interés en Microsoft .net Frameworks en el Mundo. Fuente https://www.google.es/trends

Fig. 35. Interés en Microsoft .net Frameworks por Países. Fuente

https://www.google.es/trends

Fig. 36. Interés en Microsoft .net Frameworks en Colombia. Fuente

https://www.google.es/trends

Acorde con las practicas planteadas desde 2011 hasta 2016 se ha estabilizado y madurado en Framework para Java, por lo que se hace necesario, un desarrollo de .net Framework o Microsoft .net Frameworks por el interes de la población de programadores que ha estado oscilando entre un 25% y un 50% en los años 2014 y 2015.

REFERENCIAS

[1] Hurtado, J. Desarrollo de Software basado en Componentes en la Plataforma J2EE. Popayan, Cauca.
[2] Andalucia, J. d. Marco de Desarrollio de la Junta de Andalucia. Recuperado el 1 de 2 de 2012, de http://www.juntadeandalucia.es/servicios/madeja/glossary/15/letterj