

**TITULACIÓN DE SISTEMAS INFORMÁTICOS Y COMPUTACION**

**PROCESOS DE INGENIERÍA DE SOFTWARE**

**PROYECTO**

**SEGUNDO BIMESTRE**

**Nombre:**

Carlos Caraguay

**Tutores:**

Ing. Daniel Guamán

**Fecha:** 15 – 06 – 2015

**Periodo Académico** Abril – Agosto 2015

UTPL

2015

**UNIVERSIDAD TECNICA PARTICULAR DE LOJA**

**ARQUITECTURA A UTILIZAR:**

**Arquitectura Cliente – Servidor**

La arquitectura cliente-servidor es un modelo de aplicación distribuida en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, quien le da respuesta. Esta idea también se puede aplicar a programas que se ejecutan sobre una sola computadora, aunque es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras.



**Fig. 1 Arquitectura cliente – servidor**

**Cliente**

El cliente es el proceso que permite al usuario formular los requerimientos y pasarlos al servidor, se le conoce con el término front-end.

El Cliente normalmente maneja todas las funciones relacionadas con la manipulación y despliegue de datos, por lo que están desarrollados sobre plataformas que permiten construir interfaces gráficas de usuario (GUI), además de acceder a los servicios distribuidos en cualquier parte de una red.

Las funciones que lleva a cabo el proceso cliente se resumen en los siguientes puntos:

* Administrar la interfaz de usuario.
* Interactuar con el usuario.
* Procesar la lógica de la aplicación y hacer validaciones locales.
* Generar requerimientos de bases de datos.
* Recibir resultados del servidor.
* Formatear resultados.

**Servidor**

Es el proceso encargado de atender a múltiples clientes que hacen peticiones de algún recurso administrado por él. Al proceso servidor se le conoce con el término back-end. El servidor normalmente maneja todas las funciones relacionadas con la mayoría de las reglas del negocio y los recursos de datos.

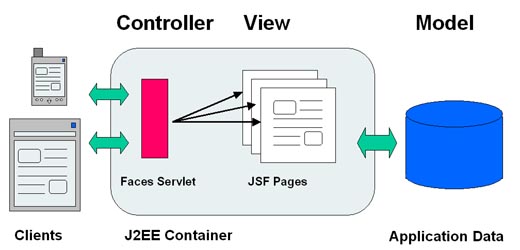
Las funciones que lleva a cabo el proceso servidor se resumen en los siguientes puntos:

* Aceptar los requerimientos de bases de datos que hacen los clientes.
* Procesar requerimientos de bases de datos.
* Formatear datos para trasmitirlos a los clientes.
* Procesar la lógica de la aplicación y realizar validaciones a nivel de bases de datos.

**PATRÓN DE DISEÑO A UTILIZAR:**

**MVC – Modelo vista controlador**

El patrón utilizado es a la vez es una Web estándar de interfaz de usuario de Java, así como un marco de trabajo que sigue firmemente el patrón de diseño Modelo-Vista-Controlador (MVC). Esto hace que las aplicaciones mucho más manejable porque el código de interfaz de usuario (Vista) se separa limpiamente de los datos de la aplicación y la lógica ( Modelo ). Para preparar el marco JSF, que proporciona acceso a los datos de aplicación a las páginas, y para evitar el acceso no autorizado o inadecuado de las páginas, todas las interacciones del usuario con la aplicación son manejados por un front-end "Faces" servlet ( Controller ).



**Fig. 2. MVC**

**Modelo**

Representa al estado de la aplicación. Puede haber dos opciones esencialmente:

* Struts proporciona una clase base org.apache.struts.action.ActionForm que se debe extender cuando se desea obtener la entrada de datos proporcionada por el usuario en la petición HTTP.
* El modelo puede ser un Bean o clase ordinaria sin necesidad extender ActionForm.

**Vista**

La vista es una página JSP que no debe contener lógica de negocio, ni flujo de la aplicación e información del modelo, sólo tags. Utiliza el modelo generado para obtener la información y presentarla

**Controlador**

El Servlet ActionServlet actúa de controlador, recibe la petición del navegador y decide qué subclase de Action va tratar la petición en función de lo que se ha declarado en el fichero de configuración struts-config.xml.

Subclase de Action. Actualiza el estado del modelo, y, controla el flujo de la aplicación y tratamiento de errores. Una instancia de una subclase de Action puede tratar la petición y responder al cliente o indicar al Servlet controlador a qué componente del sistema debe delegar el control (esta es la opción la que se lleve a cabo). Las instancias de las subclases de Action tienen acceso al contexto del Servlet controlador y demás objetos que actúan con el contenedor Web.

**HERRAMIENTA PARA EL CONTROL DE VERSIONES A UTILIZAR:**

Para el control de versiones de software se va ha utilizar la herramienta denominada **github**, la misma que es una plataforma de desarrollo colaborativo de software que permite alojar proyectos utilizando el sistema de control de versiones Git.