Manual de Usuario
Sistema de Información SICIUD

Módulo de Administración de Cuentas y Servicios

Tabla de contenido

Tabla de Ilustraciones

1) Estructura General del SICIUD:

El sistema de información cuenta con una estructura de 3 capas las cuales se encuentran descritas en la siguiente imagen:

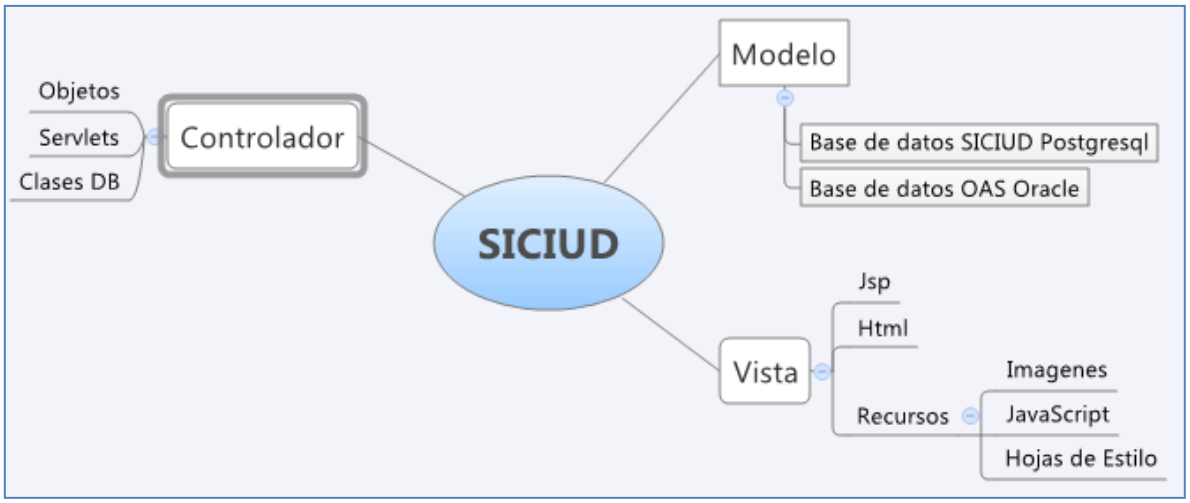


Ilustración 1: Estructura General SICIUD

1.1. Estructura de paquetes:

El sistema de información cuenta con varios módulos, los cuales tienen la estructura de la siguiente imagen:

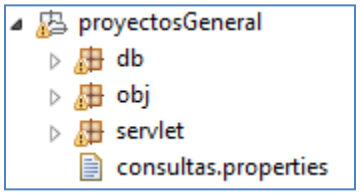


Ilustración 2: Estructura de Paquetes Java

Esta estructura es la que se debe mantener para cada módulo que se desea crear o actualizar. Dentro de estos paquetes deben estar almacenadas las clases para la gestión de toda la lógica del negocio.

1.2. Controlador:

La parte de Controlador del sistema SICIUD es el conjunto de clases encargadas de realizar toda la lógica del negocio, validaciones y gestión de los recursos de la base de datos. Estas clases deben estar empaquetadas de acuerdo a su función, por ejemplo:

→ Clases de Transporte (Obj):

Estas clases son las encargadas de hacer el encapsulamiento de la información que se envía desde y hacia las capas de Vista y Modelo. Deben estar empaquetadas en la carpeta llamada OBJ del paquete del módulo.

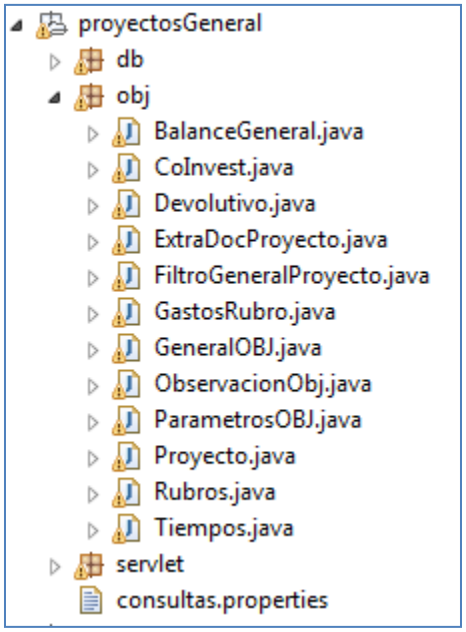


Ilustración 3: Paquete Obj

→Clases de Control (Servlet):

Estas clases son las encargadas de hacer el control del flujo en la navegación del usuario y el transporte de la información que viaja entre las 3 capas mediante los objetos (beans) mencionados en el ítem anterior.

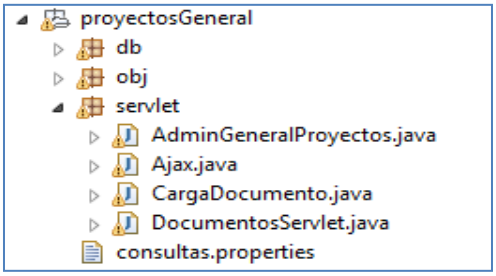


Ilustración 4: Paquete Control

→Clases de Negocio (DB):

Estas clases son las encargadas de hacer todos los procedimientos de almacenamiento, consulta, actualización y eliminación de registros sobre la base de datos. Adicional a las clases Java, se debe tener un archivo de texto plano con extensión .properties con en el cual se deben almacenar todas las sentencias SQL que serán usadas en las clases DB (las sentencias SQL NO DEBEN SER QUEMADAS EN LA CLASE).

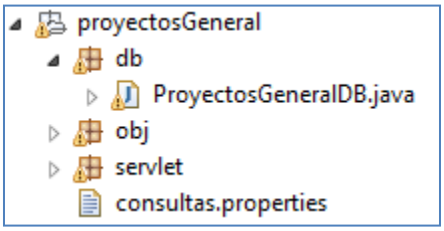


Ilustración 5: Clases de Negocio

1.3. Vista:

La parte de la Vista en el sistema SICIUD está compuesta por varios elementos encargados de únicamente imprimir la información que se le presenta al usuario, para ello es necesario el uso de elementos como las páginas estáticas (html), páginas dinámicas (jsp) y complementos como imágenes, hojas de estilo, Archivos JavaScript y demás elementos que se requiera para el diseño de la impresión y visualización.

Por otra parte, el sistema debe tener organizadas esas páginas en paquetes o carpetas etiquetadas con el nombre del módulo al cual pertenece. Únicamente deben estar en esos paquetes las páginas html y jsp que vayan a ser usadas por el respectivo módulo; en cuanto a las imágenes y demás recursos deben ser almacenadas en el paquete general encargado de centralizar todos los recursos visuales del sistema.

A continuación se encuentra un esquema de la estructura que debe tener la capa de Vista en el sistema SICIUD:

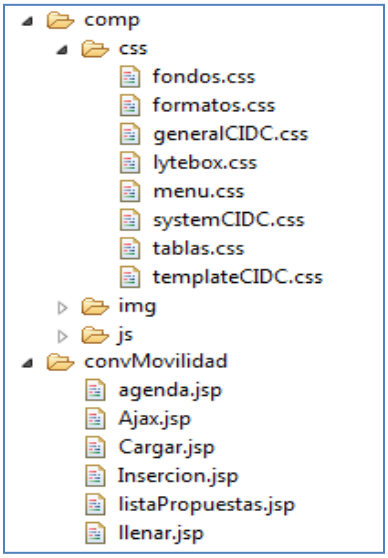


Ilustración 6: Estructura Vista SICIUD

2) Recursos Base del SICIUD:

El sistema de Información tiene una estructura interna compuesta de clases y páginas que nos permiten reutilizar y organizar el código que se va a desarrollar para todos los módulos del sistema SICIUD. La siguiente es la descripción de los recursos que no deben ser modificados, ya que son instanciados desde todos los módulos del sistema:

2.1 Clases:

- `public class baseDB:`

Esta clase es instanciada por todas las clases de Negocio ya que estas Heredan sus métodos para establecer y cerrar conexiones; además es la encargada de hacer las impresiones de los errores que se generan en tiempo de ejecución.

Los métodos más importantes para esta clase son:

```
public String getMensaje()

public void cerrar(Connection cn)

public void cerrar(PreparedStatement pst)

public void cerrar(ResultSet rs)

public void lanzaExcepcion(Exception e)

public void lanzaExcepcion(SQLException e)

public void lanzaExcepcion(MessagingException e)
```

```
public void lanzaExcepcion(AddressException e)

public void lanzaExcepcion(FileUploadException e)
```

- `public class CursorDB`

Esta clase se encarga de establecer la conexión con la base de datos en Postgresql y también con la base de Condor (oracle). Esta clase es instanciada en cada servlet para enviar una instancia de este objeto a la clase de negocio que hará la respectiva acción sobre los registros de la base de datos. Esta clase hace uso de la información contenida en el archivo Properties donde se encuentran los datos de conexión a la base de datos.

Los métodos más importantes para esta clase son:

```
public Connection getConnection(int id) throws Exception

public Connection abrir() throws Exception

public String getUrl()

public Connection abrirOracle()
```

- `public class ServletGeneral extends HttpServlet`

Este servlet es una clase que es heredada por todos los servlet que se desarrollan en el sistema SICIUD para hacer el control del flujo de navegación en el sistema. Esta clase nos permite hacer un acceso al servlet por medio de los dos métodos de acceso (GET, POST).

Los métodos más importantes para esta clase son:

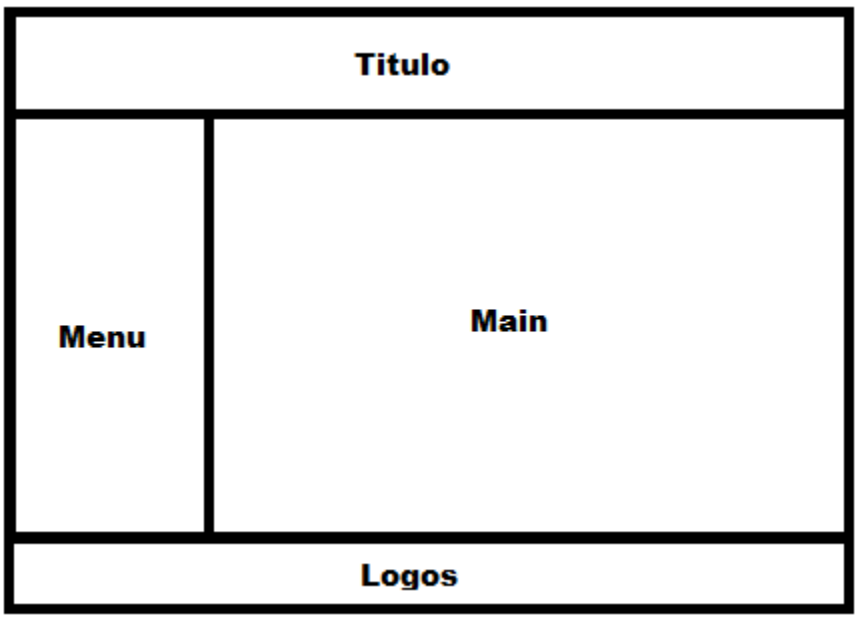
```
public void init(ServletConfig config) throws ServletException

public void doGet(HttpServletRequest req, HttpServletResponse resp)

public void doPost(HttpServletRequest req, HttpServletResponse
resp)
      throws ServletException,
      IOException
```

2.2 Páginas:

El sistema de información SICIUD tiene una estructura de frames comandada por la página index.html, la cual tiene en su cuerpo la estructura de los frames del sistema.



Todas las páginas que se desarrollan para los módulos del sistema SICIDU son visibles únicamente para el frame **MAIN.** Los demás frames están dedicados a la visibilidad de los logos y titulo del portal. El marco de la parte izquierda está encargado de mostrar el menú.

Por otra parte, todas las páginas JSP hacen importación de una página genérica que contiene el código que se repite en todas las JSP y html, tal como la importación de las hojas de estilo, los archivos de javascript y el código que permite hacer la impresión de los mensajes informativos. Esta página se llama **General.jsp** y es importada mediante la siguiente instrucción.

```
<c:import url="/general.jsp" />
```

3) Pasos para nuevo módulo en SICIUD:

Para el desarrollo de un módulo nuevo en el sistema SICIUD, se debe tener en cuenta la estructura de los puntos 1.1, 1.2 y 1.3.El orden de desarrollo puede ser (no obligatorio) el siguiente:

3.1. Desarrollar Objetos de Transporte:

Estas clases son las encargadas de hacer el transporte de los datos entre las clases de Negocio, los servlets y las JSP. Su estructura es bastante simple y consisten únicamente en un conjunto de atributos que representan los parámetros o características de una entidad y con sus respectivos métodos GET y SET.

La siguiente es la representación de esa clase:

```
public class ObjetoEjemplo {

        private String atributoUno;
        private String atributoDos;
        private String atributoTres;

        public String getAtributoUno() {
            return atributoUno;
        }
        public void setAtributoUno(String atributoUno) {
            this.atributoUno = atributoUno;
        }
        public String getAtributoDos() {
            return atributoDos;
        }
        public void setAtributoDos(String atributoDos) {
            this.atributoDos = atributoDos;
        }
        public String getAtributoTres() {
            return atributoTres;
        }
        public void setAtributoTres(String atributoTres) {
            this.atributoTres = atributoTres;
        }

}
```

Ilustración 7: Clase Objeto Transporte

3.2. Desarrollar Clases de Negocio:

Estas clases deben tener la siguiente estructura en la cual se van a desarrollar todos los métodos necesarios para la gestión de la información en la base de datos que el módulo va a manipular.

```
import java.util.ArrayList;
import java.util.List;
import cidc.general.db.BaseDB;
import cidc.general.db.CursorDB;
import cidc.proyectos.obj.ObjetoEjemplo;

public class ClaseNuevaEjm extends BaseDB {

    public ClaseNuevaEjm(CursorDB c, int p) {
        super(c, p);
    }
    public boolean insertar(ObjetoEjemplo obj){
        boolean retorno=false;
        return retorno;
    }
    public ObjetoEjemplo consultarElemento(ObjetoEjemplo obj){
        ObjetoEjemplo retorno=null;
        return retorno;
    }
    public List<ObjetoEjemplo> consulta(ObjetoEjemplo obj){
        List<ObjetoEjemplo> retorno=new ArrayList<ObjetoEjemplo>();
        return retorno;
    }
    public boolean eliminar(ObjetoEjemplo obj){
        boolean retorno=false;
        return retorno;
    }
    public boolean actualizar(ObjetoEjemplo obj){
        boolean retorno=false;
        return retorno;
    }
}
```

Ilustración 8: Clase de Negocio plantilla

Esta clase debe contener por lo menos los métodos básicos de Inserción, Consulta, Actualización y eliminación. Los 3 métodos de eliminación, inserción y actualización deben retornar una variable booleana la cual indicará si se ha hecho o no satisfactoriamente la acción solicitada; los otros dos métodos deben retornar el respectivo elemento de consulta o la lista de registros encapsulados en el objeto de transporte que corresponde.

Adicional a estos métodos se pueden incluir los demás métodos que sean necesarios para el funcionamiento del módulo en el sistema SICIUD.

3.3. Desarrollar Clase de Control (Servlet):

Estas clases son las encargadas de hacer el control del flujo de navegación y hacer la entrega de los objetos desde y hacia las clases del negocio y las JSP.

La estructura interna consiste en un método llamado **OPERACIONES** el cual contiene un bloque **SWITCH** en donde se ejecutarán cada una de las posibles llamadas al servlet. En otras palabras, este switch nos permite organizar (en cada uno de los **CASE**) todas las acciones que el módulo requiera; por ejm cada case puede contener una acción como el llamado al método de inserción, el llamado al método búsqueda, el llamado al método consulta, etc; y los demás métodos que se encuentran programados en las clases de negocio.

La siguiente imagen muestra una plantilla de cómo deben ser todos los servlets de cada módulo.

```
public class ServletEjemplo extends ServletGeneral {

    public String [] operaciones(HttpServletRequest req, HttpServletResponse resp)
    throws ServletException, IOException {
        context=config.getServletContext();
        cursor=new CursorDB();
        HttpSession sesion=req.getSession();
        String irA="/modulo/PaginaPrincipal.jsp";
        Usuario usuario=(Usuario)sesion.getAttribute("loginUsuario");
        ClaseNuevaEjm nuevaClaseDB=new ClaseNuevaEjm(cursor,usuario.getPerfil());
        mensaje="";
        int accion=0;
        if(req.getParameter("accion")!=null)
            accion=Integer.parseInt(req.getParameter("accion"));
        if(req.getAttribute("accion")!=null && accion==0)
                accion=Parametros.cmdContratoPdf;
        retorno[0]="unir";
        switch(accion){
            case 1:
                irA="/modulo/PaginaDeInsercion.jsp";
            break;
            case 2:
                irA="/modulo/PaginaConsulta.jsp";
            break;
            case 3:
            default:
                irA="/modulo/PaginaPrincipal.jsp";
             break;
        }
        accion=0;
        retorno[1]=irA;
        retorno[2]=mensaje;
        return retorno;
    }
```

Se debe tener en cuenta que estas clases heredan de la clase GeneralServlet y esta a su vez de HttpServlet, por tal motivo no es necesario definir los métodos INIT, DOPOST y DOGET. Ya que estos están definidos en la clase **SERVLETGENERAL.**

Adicional a la estructura mostrada anteriormente para los servlet, el desarrollador puede agregar los métodos que se consideren necesarios para complementar la funcionalidad requerida para el módulo.

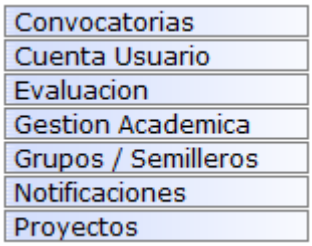
4) Pasos para crear un nuevo ítem en el menú principal:

El menú se encuentra formado por la información obtenida de la base de datos, ya que este muestra dinámicamente los módulos que están asignados a cada perfil en el sistema.

Las siguientes tablas son afectadas al momento de crear un nuevo elemento en el menú principal:

- **b_categoria**

  Esta tabla contiene el nombre de los ítems superiores del menú principal. Estos ítems son aquellos que tienen la siguiente apariencia:

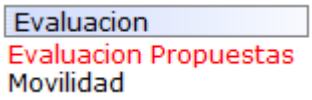
  

  En esta tabla solamente debe ser llenada con el nombre del Item superior y el número consecutivo; la siguiente es una sentencia SQL que registra un nuevo elemento en la tabla:

  insert into b_categoria (bcatid,bcatnombre)values(00,Nombre Categoria);

- **b_item_categoria**

  Esta tabla contiene el nombre del subitem asociado a una categoría en el menú principal; estos sub ítems tienen la siguiente apariencia:



Esta tabla debe ser llenada con la llave foránea del código de la categoría a la que pertenece, el nombre del sub ítem y el código del perfil principal que tendrá permiso sobre este ítem. La siguiente es una sentencia SQL que registra un nuevo elemento en la tabla:

insert into b_item_categoria (biccategoria,bicnombre,bicperfil)values(00,'Sub Item Categoría,1);

- **recursos**

  Esta tabla contiene el listado completo de direcciones URL que pueden ser accedidas en el sistema SICIUD; Es necesario mapear todas las URL en esta tabla con el fin de poder habilitar o restringir los permisos de acceso a los módulos del sistema.

  Cada Item registrado en la tabla **b_item_categoria** debe tener asociados uno o más registros en la tabla **recursos** y uno de esos recursos debe tener como valor TRUE los campos rprimario y rvisible. De este modo se está asociando una URL al subitem de una categoría. La siguiente es una sentencia SQL que registra un nuevo elemento en la tabla:

  insert into recursos
  (rnombre,rurl,riditem,ridperfil,rprimario,rvisible)values('Informes','/Informes/AdminInformes.x',61,1,true,true);

- **Item_perfil**

  Esta tabla contiene la relación de permisos asociados entre los perfiles y los ítems de categorías. Cada Item puede estar relacionado a uno o más perfiles. La información aquí contenida es administrada desde el módulo de gestión de servicios del SICIUD. La siguiente es una sentencia SQL que registra un nuevo elemento en la tabla:

  insert into item_perfil(ipiditem,ipidperfil)values(61,1);

Una vez se hayan registrado todos los datos en las tablas mencionadas anteriormente, el menú podrá ser administrado desde el módulo de servicios del SICIUD; Adicionalmente no podrán haber ingresos a páginas o recursos que no se encuentren mapeados debido a que el filtro (clase administrada por el servidor tomcat) validará cada url de acceso para permitir o denegar acceso a los usuarios que hacen uso del aplicativo.