Capitulo IV. Desarrollo de Software

En base al esquema de base de datos diseñado para este sistema, mostrado en el capitulo anterior, y en base a los diagramas construidos bajo el modelado de la metodología UML, presentado en el capítulo anterior, lo que se construyeron clases Java, Java Beans, Java JSPs y Java Servlets en la versión 1.3 de Java.

Desde un principio se construyo la aplicación de acuerdo al diseño de la aplicación, el cual esta integrado por el BackEnd y el FrontEnd, en un principio se hicieron algunos cambios dentro de la base de datos, ya que varios atributos de la tabla pg_articulos fueron modificados debido a que algunos campos no eran estrictamente necesarios.

Otro cambio hecho en el primer diseño del sistema fue que en la parte del FrontEnd en la cual se pretendía utilizar tres beans, uno de ellos para controlar las peticiones y adminístralas, otro para administrar las secciones del sitio y un último Bean para administrar los artículos del sitio. Pero en esta fase de desarrollo fue mejor programar un solo Bean el cual administrara tanto las peticiones de los usuario, como el desplegar los artículo y secciones necesarias para la navegación del sitio, es decir, este componente se encargará de desplegar el sitio, permitir que sea navegable y además controlar la concurrencia de usuarios y el acceso a cada uno de las secciones y artículos.

Para el desarrollo del sistemas son necesarias las librerías de java tanto para crear clases como para crear servlets y jsp, pero también se necesitan de la librería de Java para

interactuar con el lenguaje Perl, ya que por medio de este lenguaje se identificarán diferentes directivas y se crearán variables propias para cada sitio creado con el Módulos Administrativo de Sitios Interactivos para Web. Otra librería necesaria es la librería de O´Reilly la cual nos permite tomar archivos y desde la maquina del usuario y colocarlos en el servidor de la aplicación.

IV.1. Back End y Front End.

Como se explico en el capitulo anterior este proyecto fue desarrollado en dos partes, el Back End (parte administrativa) y el Front End (producto final del sistema), cada una de estas partes fue desarrollada bajo la tecnología "Model View Controller".

El Back End, que es la parte administrativa para construir el Front End, consta de varios métodos como lo son los de conexión con la base de datos, y los métodos propios para agregar secciones, artículos, templates y noticias dentro de la base de datos, además que cuenta con otra clase que se encarga de agregar el texto, las imágenes y archivos que se vayan a utilizar dentro del Front End.

A continuación se explicará toda la parte referente Back End, en primera instancia el Model, es decir, la lógica del manejo de información, posteriormente se explicará lo referente al Controller (control de información entre procesamiento de información y presentación de información) y finalmente de mostrará el View (presentación de información).

En la siguiente tabla se muestran las clases desarrolladas para el manejo de la base datos, esta tabla contiene una pequeña descripción de cada una de las clases para el manejo y configuración de la base de datos.

IV.1.1. Base de Datos.

Nombre	**Descripción**
ConnectionPool	Esta es la clase encargada de realizar la conexión con la base de datos.
DBResults	Almacena el resultado de un query dentro de vector de Strings. Contiene un método para conocer los atributos de una base la base de datos utilizada.
DataBaseUtilities	Esta clase se encarga de ejecutar un query y colocar los resultados dentro de un Objeto. Además permite crear una nueva tabla en la base de datos; así como también contiene un método para imprimir todos los datos de una tabla de la base de datos.
DriverUtilities	Clase encargada de proporcionar el URL para el driver de la base de datos utilizada, cabe mencionar que esta clase no proporciona el driver.
GnaSvtPortalDB	Es la clase encargada de guardar los parámetros de la base de datos utilizada. Estos parámetros son guardados dentro de un archivo de configuración colocado en la carpeta de la aplicación, y llamado "`servletPortal.cfg`"

Tabla 1. Módulos programados para el manejo de Base de Datos.

Las clases más importantes para el manejo de la base de datos son la clase ConnectionPool y la clase GnaSvtPortalDB, ya que la primera de ellas es la encargada de abrir la conexión con la base de datos utilizada por el sistema y monitorear cuantas conexiones están activas y cuantas están inactivas.

La clase ConnectionPool contiene los siguientes atributos:

private String driver, url, username, password;

private int maxConnections;

private boolean waitIfBusy;

private Vector availableConnections, busyConnections;

private boolean connectionPending = false;

La primera línea contiene los atributos correspondientes de la base de datos, lo que se llama "DataBaseMetaData", los siguientes atributos son para conocer el número máximo de conexiones que se abrirán, el availableConnections se refiere a las conexiones libres, y busyConnections a las conexiones ocupadas.

Dentro del siguiente diagrama de clase podemos ver todos aquellos métodos que existen dentro de esta clase

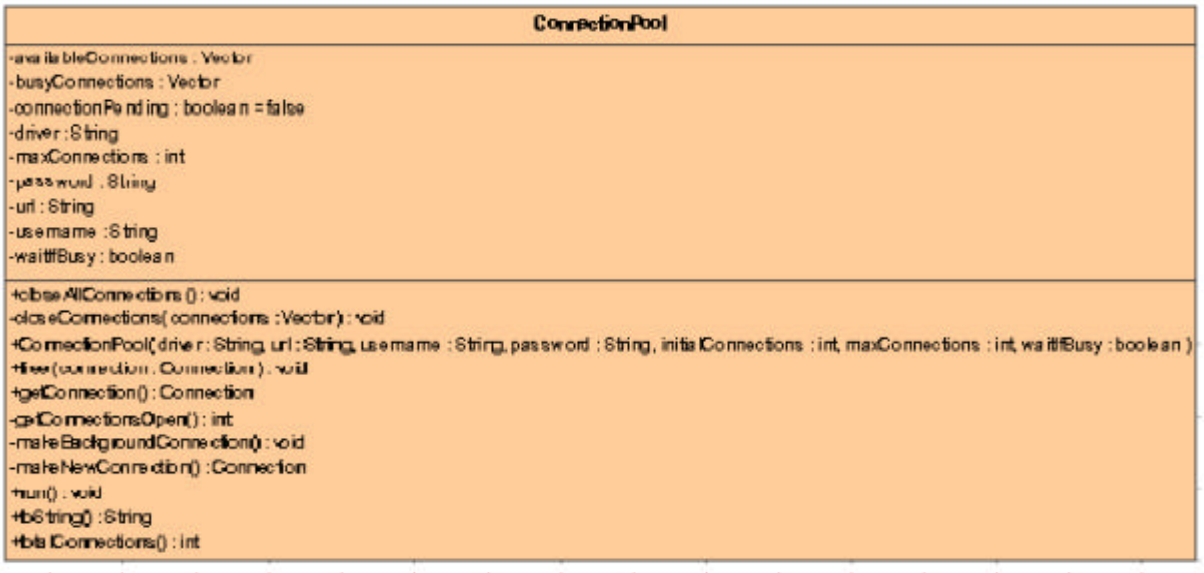

Figura 16. Diagrama de Clase de ConnectionPool.

Otra clase muy importante es la clase GnaSvtPortalDB, ya que es la encargada de registrar las características de la base de datos que será utilizada dentro del sistema, estas características son las que se muestran en la siguiente pantalla:

Figura 17. Pantalla del Back End para el módulo de Base de Datos.

Una vez que se colocan los datos correctamente esta pantalla utiliza al Servlet GnaSvtPortalDB para tomar todos los datos y colocarlos de un archivo de configuración llamado "`servletPortal.cfg`", el cual esta situado en la carpeta Web-Inf de la aplicación. Cabe mencionar que este archivo es encriptado, por lo cual la única manera de modificarlo es entrando a la opción Base de Datos del Módulo Administrador de Contenidos para Portales Interactivos Web.

La siguiente figura es el diagrama de clase correspondiente a la clase GnaSvtPortal DB:

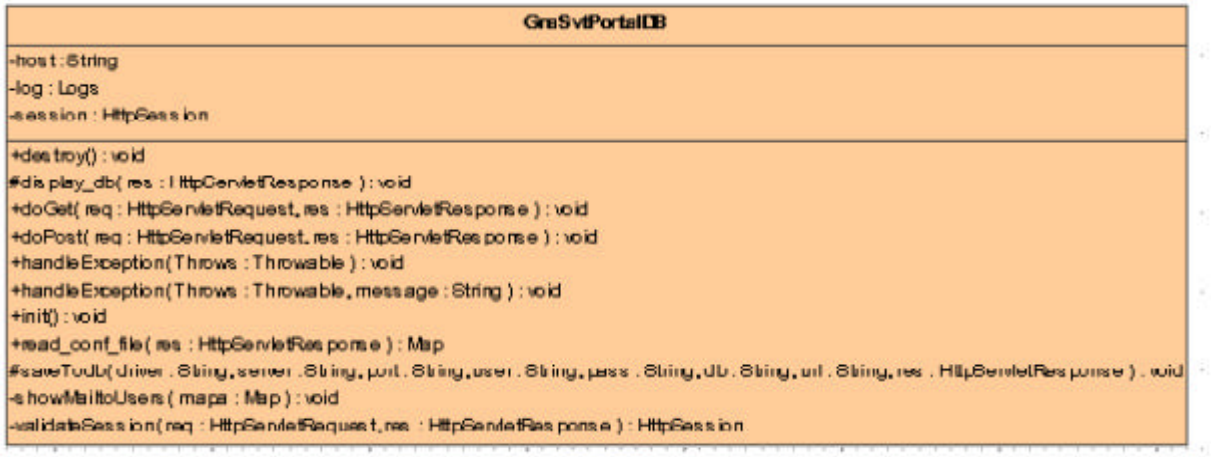

Figura 18. Diagrama de clase para GnaSvtPortalDB.

Los métodos más importantes dentro de esta clase son saveTodb(), ya que como se explico anteriormente toma todos los datos correspondientes a la base de datos, los encripta y los guarda dentro de un archivo. El otro método importante es el método read_conf_file (), ya que este es el encargado de decodificar el archivo que contiene la información d el base de datos, la cual es utilizada por el método display_db () para mostrar al usuario la configuración de la base datos.

Ahora bien vamos a explicar de aquellas clases que son las encargadas de construir el Front End, es decir las clases encargadas de almacenar la información en la base de datos, que posteriormente es tomada por los Java Beans para crear el sitio. Al igual que para las clases encargadas de la base de datos, mostraremos una tabla con la descripción de las clases de Secciones, Artículos, Templates, Noticias y la clase de Upload.

Módulos de Aplicación.

Nombre	Descripción
GnaSvtPortalArt	Es la clase encargada de la administración de los Artículos de información.
GnaSvtPortalNoticias	Es la clase encargada de la administración de Noticias, las cuales son manejadas como artículos de información
GnaSvtPortalSec	Es la clase encargada de la administración de las Secciones de Información.
GnaSvtPortalTemplate	Es la clase encargada de la administración de Templates para ser utilizados por los artículos y por las Secciones.
Upload	Esta clase es utilizada por GnaSvtPortalTemplate, GnaSvtPortalSec y GnaSvtPortalArt para agregar el texto, imágenes y archivos correspondientes a cada uno.

Tabla 2. Descripción de los módulos centrales del Back End.

IV.1.2. Templates.

La clase GnaSvtPortalTemplate es la de mayor importancia en este bloque, ya que ella proporciona a interfaz en la que un artículo o sección se desplegará. Esta clase, además de proveer las acciones agregar, eliminar y modificar un template se encarga de tomar un archivo .zip -el cual contiene todos los archivos correspondientes al template-, descomprimirlo y colocarlo en la carpeta del servidor correspondiente a los templates.

Las tablas de la base de datos que utiliza el servlet de templates son las siguientes:

- Pg_template. Para agregar la información correspondiente al template.
- Pg_variable. Para agregar el código de las variables correspondiente a cada template.

Esta es una clase muy completa ya que utiliza el manejo de archivos, acceso con Base de Datos, utiliza el package java.util.zip.*; para descomprimir y además hace el uso de la librería O´Reilly para colocar en el servidor el snapshoot correspondiente al template.

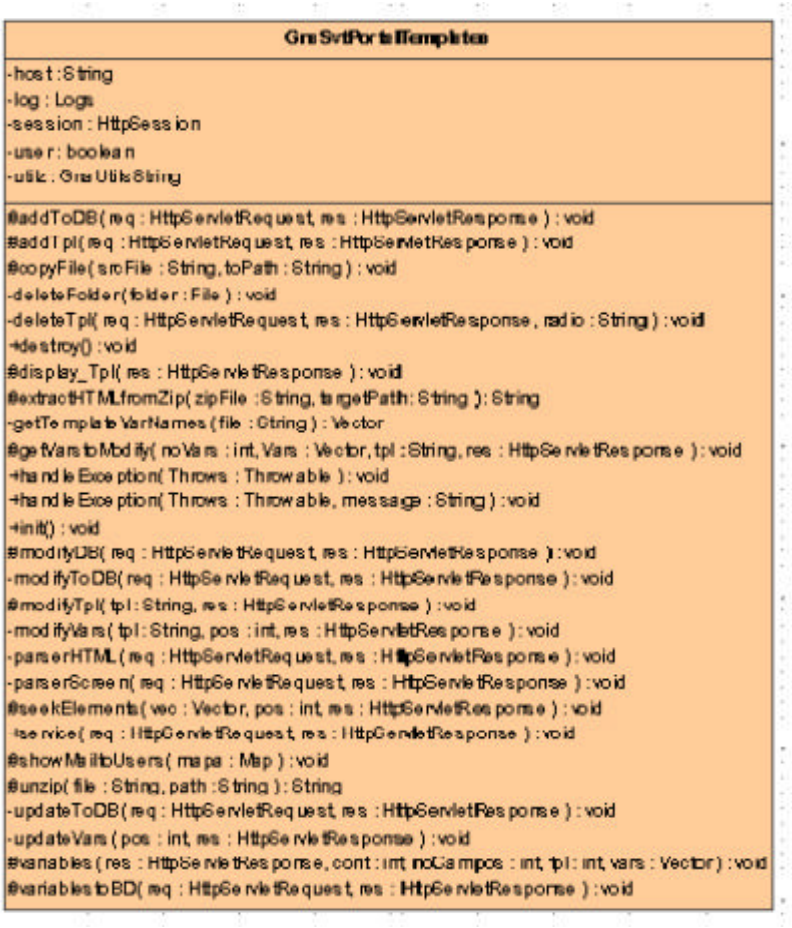

Figura 19. Diagrama de clase para GnaSvtPortalTemplate.

Como podemos observar en el diagrama de clase anterior, esta clase cuenta con muchos métodos, siendo los más importantes todos aquellos métodos para agregar un template, ya que primero se debe mostrar la pagina para agregar los datos correspondientes

a un nuevo Template por medio del método addTpl(), el cual utiliza un archivo HTML donde escribe la forma correspondiente.

Figura 20. Pantalla del BackEnd para agregar un nuevo Template.

Una vez que todos los datos son correctos, se ejecuta el método addToDB(), el cual toma todos los datos escritos en la pantalla anterior para insertarlos dentro de la base de datos, pero antes de ejecutar el query, se descomprime crea una carpeta temporal dentro del servidor. Esto se hace porque el siguiente paso es descomprimir el archivo .zip y extraerlo a esa carpeta temporal para verificar que tenga la estructura correcta, que es la siguiente:

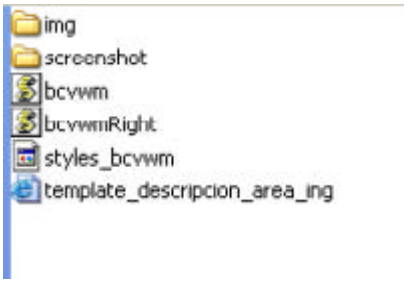

Figura 21. Diagrama de la estructura del archivo zip para un template.

En caso de que la estructura sea la correcta todos estos archivos serán puestos dentro del servidor en la carpeta correspondiente a los templates. En caso contrario se mostrará un mensaje en pantalla, indicando que el zip no es válido.

Posteriormente se lee el archivo HTML que venía dentro del .zip y se escanea hasta encontrar las variables de ese template, en caso de encontrarlas se mostrará la siguiente pantalla para cada variable:

Figura 22. Pantalla del BackEnd para agregar el código de las variables del template.

Como se muestra en la imagen es necesario escribir el código correspondiente a cada variable, cabe mencionar que dentro del manual de usuario se muestra una tabla con las variables de ambiente que existen por default en el sistema. El siguiente código es el método encargado de almacenar una las variables dentro de la base de datos.

```
protected void variablestoBD(HttpServletRequest req, HttpServletResponse res)throws ServletException, IOException {
                    GnaSvtTemplate template = new GnaSvtTemplate(path);
                    Map va = new HashMap();
                    String txt = "", query, count;
                    PrintWriter out = res.getWriter();
                    int num=0;
                    boolean insert=false, error=false;

                    String variable = req.getParameter("varname");
                    String codigo = req.getParameter("code");
                    int contador= Integer.parseInt(req.getParameter("contador"));
                    int campos = Integer.parseInt(req.getParameter("noCampos"));
                    int tpl = Integer.parseInt(req.getParameter("template"));
                    String[] Vars= req.getParameterValues("vars");
                    Vector vars = new Vector();
                    for (int i=0; i<Vars.length; i++)
                      vars.add(Vars[i]);

          try {
            Connection connection = connectionPool.getConnection();
                    query = "select max(id_variable) + 1 from pg_variable";
                    DBResults results = DatabaseUtilities.getQueryResults(connection,query, false);

                            String[] data = new String[1];
                            data = results.getRow(0);
                            String id = data[0];

               query = "insert into pg_variable values("+id+", "+tpl+", '"+variable+"', '"+codigo+"')";
                    insert = DatabaseUtilities.update(connection, query);
                            contador++;

                            if (contador <= campos){
                            variables(res, contador, campos, tpl, vars);
                            }else{
                            String tmp = Servlet + "mx.com.gedas.portal.GnaSvtPortalTemplates?accion=home";
                                    res.sendRedirect(tmp);
                            }

                            connectionPool.free(connection);
                    }
                    catch(Exception e){
                            handleException(e);
                    }
          }
```

Figura 23. Código para agregar Variables de Templates en Base de Datos.

Finalmente, después de agregar las variables a la base de datos se ejecuta el query para registrar el template en la base de datos. El siguiente método es el encargado de descomprimir, validar y en caso agregar el template a la base de datos.

```
protected void addToDB(HttpServletRequest req, HttpServletResponse res)throws ServletException, IOException {
                        GnaSvtTemplate template = new GnaSvtTemplate(path);
                        Map va = new HashMap();
                        PrintWriter out = res.getWriter();
                        String query, str="";
                        int tpl;
                        boolean insert, error=false;

                        String nombre     = req.getParameter("nombre");
                        String archivo    = req.getParameter("zip");
                        String screenshot = req.getParameter("screenshot");
                        String tipo= req.getParameter("kind");

                        Connection connection = null;
            try {
              connection = connectionPool.getConnection();
              query = "select max(id_template) from pg_template";
                    DBResults results = DatabaseUtilities.getQueryResults(connection,query, false);
                                String[] data = new String[1];
                                data = results.getRow(0);
                                if (data[0] != null)
                           tpl = Integer.parseInt(data[0]);
                                else
                                  tpl = 0;
                        tpl++;

                                //create files objects
                                File zipFile    = new File(TmpPath + slash + archivo);
                                File screenFile = new File(TmpPath + slash + screenshot);

                                //create folder name to save template
                                String folderName = (tipo.equals("sec")) ? "sec" : "art";
                                folderName += "" + tpl;
                                String folder = TplPath + folderName;

                                //create target folders structure
                                File Folder1 = new File(folder + slash + "img");
                                Folder1.mkdirs();
                                File Folder2 = new File(folder + slash + "screenshot");
                                Folder2.mkdirs();

                                //unzip valid files into folder
                                String fileName = "";
                                if (zipFile.exists()){
                                  fileName = unzip(zipFile.toString(), folder);
                                }

                                copyFile(screenFile.toString(), Folder2.toString());

                                //create names to database
                                String htmlName = folderName +"/" + fileName;
                                String screenName = folderName + "/screenshot/" + screenFile.getName();

                                //delete files from temporal folder
                                zipFile.delete();
                                screenFile.delete();

                                //check template's type and count varnames
                                String tempString = folder + slash + fileName;
```

```
Vector vars = getTemplateVarNames(tempString);
Vector userVars = new Vector();
Vector sysVars = new Vector();
int campos=0, imagenes=0, varNames=0, count=0, archivos=0;
String msg = "";
for (int i=0; i<vars.size(); i++){
  String ele = vars.get(i).toString().toUpperCase();
  if ((ele.endsWith("_TXT")) || (ele.endsWith("_IMG")) || (ele.endsWith("_FILE"))){
    userVars.add(vars.get(i));
    if (ele.endsWith("_TXT"))
      campos++;
    else if (ele.endsWith("_FILE"))
      archivos++;
    else
      imagenes++;
  }else{
    sysVars.add(vars.get(i));
   varNames++;
  }
}

if (tipo.equals("sec")){
  query = "insert into pg_template values("+tpl+", '"+nombre+"', "+campos+", "+imagenes+", '"+htmlName+"', '"+screenName+"', 0)";
 insert = DatabaseUtilities.update(connection, query);
  }else{
   query = "insert into pg_template values("+tpl+", '"+nombre+"', "+campos+", "+imagenes+", '"+htmlName+"', '"+screenName+"', 1)";
 insert = DatabaseUtilities.update(connection, query);
  }

    //insert varnames to DB
    //////////////////////

    //Get id from pg_variable
    query = "select max(id_variable) from pg_variable";
 DBResults results_var = DatabaseUtilities.getQueryResults(connection, query, false);
    String[] data_var = new String[1];
    int id_var;
    data_var = results_var.getRow(0);
    if (data_var[0] != null)
  id_var = Integer.parseInt(data_var[0]);
    else
      id_var = 0;
 id_var++;

    //insert variables
    for (int i=0; i<userVars.size(); i++){
      String ele = (String)userVars.elementAt(i);
        query = "insert into pg_variable values("+id_var+", "+tpl+", '"+ele+"', '')";
 insert = DatabaseUtilities.update(connection, query);
        id_var++;
    }

    //display teplate's home
    if (sysVars.size() == 0)
        display_Tpl(res);
    else
        variables(res, 1, varNames, tpl, sysVars);
```

```
                }catch (Exception e) {
                                handleException(e, "Error, could not insert template data in database.");
                                error = true;
        }finally{
                                try{
                connectionPool.free(connection);
                                }catch(SQLException sql){
                                        handleException(sql, "Error, could not close connection");
                                }

                        if (error){
                                 str += "<INPUT TYPE=HIDDEN NAME=\"accion\" VALUE=\"home\">\n";
                                 str += "<TR>\n";
                                  str += "  <TD COLSPAN=4><P ALIGN=JUSTIFY>Error, could not insert
template data in database.</P>\n";
                                        str += "<TR>\n";
                                        str += "  <TD COLSPAN=4 HEIGHT=100 VALIGN=BOTTOM
ALIGN=CENTER><INPUT TYPE=SUBMIT VALUE=\"Continue\" CLASS=\"button\">\n";

                                va.put("CONTENT", str);
                                va.put("TITLE", "Error, invalid template's type");
                                template.set_vars(va);
                                String parse = template.print_template("template_template.html");
                                        if (parse == null)
                                out.println(template.last_error);
                                else
                                out.println(parse);

                                out.flush();
                                out.close();
                        }
                        }
                }
```

Figura 24. Código para descomprir archivo.

Es muy importante mencionar que la clase GnaSvtPortalTemplate, no es la encargada de subir las imágenes al servidor, quien se encarga de hacer este trabajo es la clase Upload, la cual cmo caracteristica utiliza la librería O´Reilly para tomar el archivo y depositarlo en el servidor. El diagrama de clase que le corresponde es el siguiente:

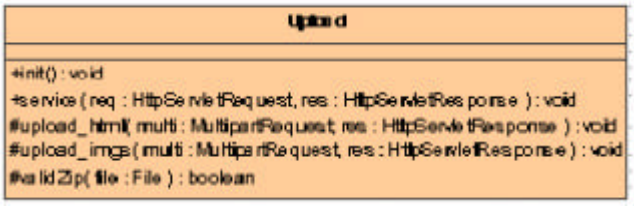

Figura 25. Diagrama de clase para Upload

Dentro del código correspondiente a esta clase podemos ver como recibe como parametro un objeto llamado multi, el cual es quien tiene relación con la librería O´Reilly.

```
protected void upload_imgs (MultipartRequest multi, HttpServletResponse res)throws ServletException, IOException {
                GnaSvtTemplate template = new GnaSvtTemplate(path);
                Map va = new HashMap();
                String query, str="";
                PrintWriter out = res.getWriter();

          String nombre = multi.getFilesystemName("files");
                File screen = multi.getFile("files");

                ImageIcon icon = new ImageIcon(screen.toString());
                int height = icon.getIconHeight();
                int width  = icon.getIconWidth();

                if (width == 300 && height == 200){
                 str += "<INPUT TYPE=HIDDEN NAME=file VALUE=\""+nombre+"\">\n";
                 str += "Image was loaded sucessfull.";
                }else{
                 str += "Error, image file must have 300 x 200 pixels.";
                 screen.delete();
                }
                va.put("CONTENT", str);

          template.set_vars(va);
          String parse = template.print_template("cargar_imgScreenShot.html");
                if (parse == null)
          out.println(template.last_error);
          else
            out.println(parse);

         out.flush();
          out.close();
}
```

Figura 26. Código cargar imágenes al servidor desde una máquina cliente..

Para terminar con la clase administradora de templates tenemos que mencionar que para modificar un template se utiliza el método updateToDB (), el cual se hace un proceso muy similar al de agregar un nuevo template, pero si se modifica el zip, se encarga de borrar el anterior.

IV.1.3. Secciones

Por parte de la clase encargada de las Secciones se utiliza el servlet llamado GnaSvtPortalSec, que además de agregar, eliminar y modificar secciones, también agrega, elimina y modifica categorías, que son las encargadas de mostrar la información en ciertas posiciones del template, esto tomando desde el punto de vista de una página HTML. Es decir, el template correspondiente a una sección puede tener el siguiente formato:

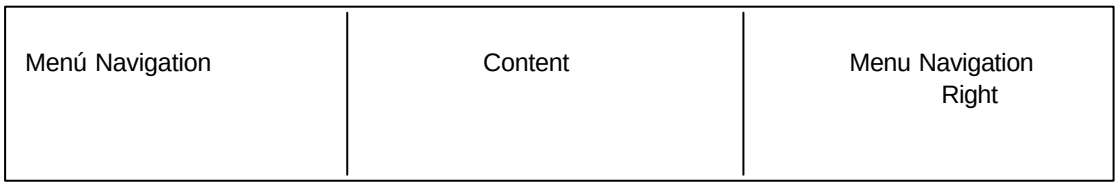

Figura 27. Diagrama ejemplo de categorías.

Las tablas de base de datos utilizadas por el servlet de secciones son las siguientes:

- Pg_sec. Para agregar la información correspondiente a una sección.
- Pg_cat. Para agregar la información correspondiente a una categoría.
- Pg_idioma. Para agregar el idioma de la sección.
- Pg_imagen. Para agregar las imágenes correspondientes a una sección
- Pg_texto. Para agregar el texto correspondiente d una sección.

?? Pg_template. Para asignar un template correspondiente a la sección.

?? Pg_file. Para agregar archivos necesarios de una sección.

La siguiente figura muestra el diagrama de clase correspondiente a Servlet encargado de la administración de las secciones:

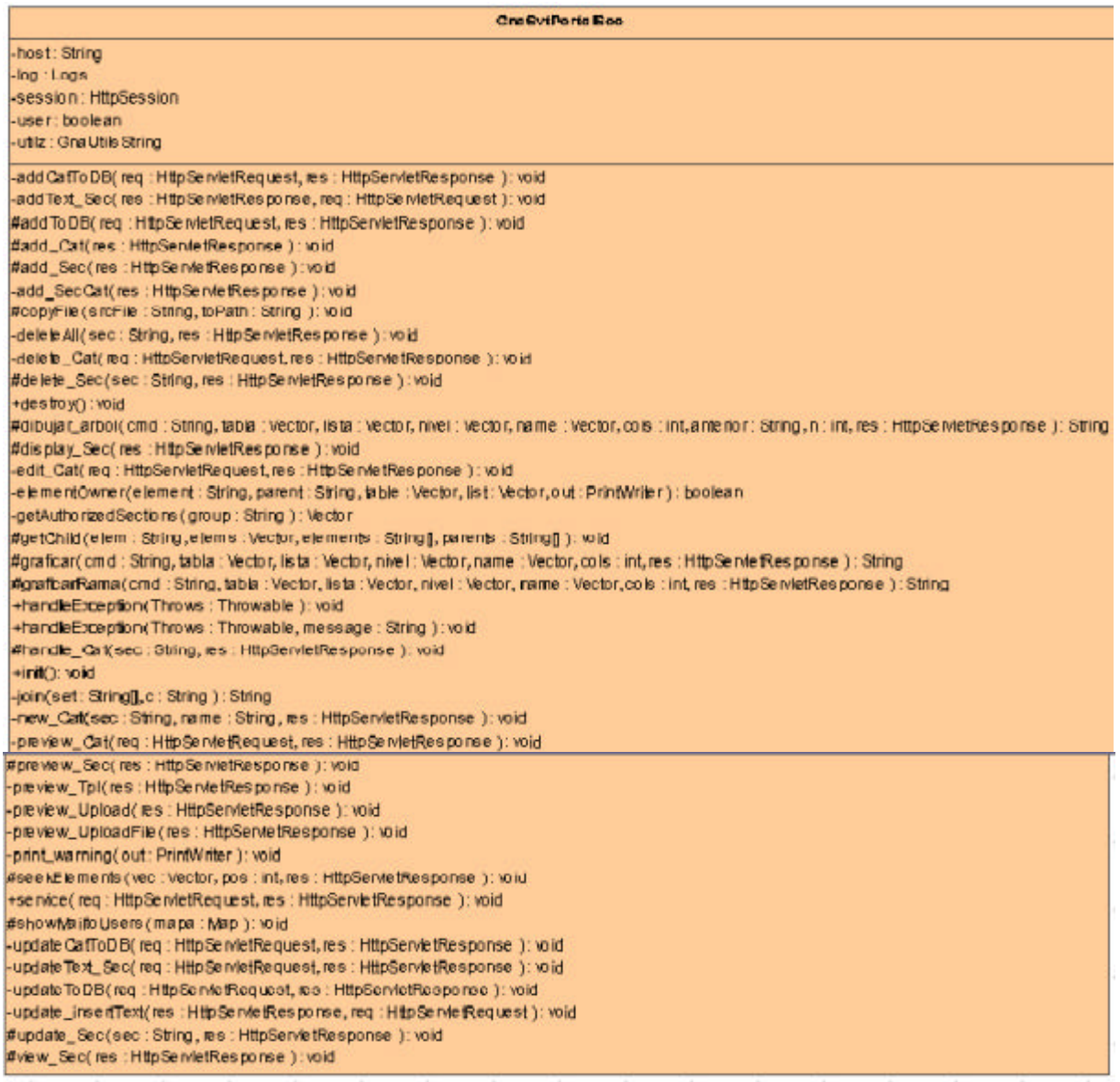

Figura 28. Diagrama de clase para GnaSvtPortalSec.

Como se menciono anteriormente se deben agregar tanto categorías, como secciones y a cada una de estas acciones le corresponde un método, el cual inserta en la tabla correspondiente en la base de datos, el código de estos métodos es el siguiente:

```
protected void addToDB(HttpServletRequest req, HttpServletResponse res)throws ServletException, IOException {
        GnaSvtTemplate template = new GnaSvtTemplate(path);
                Map va = new HashMap();
                String str="", query;
                PrintWriter out = res.getWriter();
                int id=0, level=0, idText=0, idImagen=0, idFile=0;
                boolean error = false;
                Map ve = new HashMap();
                Map vi = new HashMap();
                boolean insert;
                int count = 0;


  Enumeration params = req.getParameterNames();
  while (params.hasMoreElements()) {
   String name = (String)params.nextElement();
   String value = req.getParameter(name);
                                if (name.startsWith("GNAtexto"))
                                  va.put(name, value);
                                else if (name.startsWith("GNAimagen")){
                                        File imgFile = new File(TmpPath + slash + value);
                                        //String folder = AppPath + "articles" + slash +
                                  ve.put(name, value);
                                }else if (name.startsWith("GNAfile")){
                                        File srcFile = new File(TmpPath + slash + value);
                                  vi.put(name, value);
                                }
  }

                String name             = req.getParameter("nombre");
                String parent   = req.getParameter("parent");
                String tpl              = req.getParameter("template");
                String cat              = req.getParameter("category");
                String language         = req.getParameter("language");

                Connection connection = null;
 try {
  connection = connectionPool.getConnection();
                query = "select max(id_seccion) from pg_seccion";
                DBResults results = DatabaseUtilities.getQueryResults(connection, query, false);

        String[] data = new String[1];
          data = results.getRow(0);
                        if (data[0] != null)
                   id = Integer.parseInt(data[0]);
                        else
                          id = 0;
  id++;
```

```
                    if (parent == null || parent.equals("null"))
                     level = 1;
                     else{
                     query = "select nivel+1 from pg_seccion where id_seccion="+parent;
                    results = DatabaseUtilities.getQueryResults(connection,query, false);
                     int rows = results.getRowCount();

                    for (int i=0; i<rows; i++){
                       data = results.getRow(i);
                     level = Integer.parseInt(data[0]);
                     }
                     }

    query = "insert into pg_seccion values("+id+", "+parent+", "+tpl+", "+language+", '"+name+"', "+level+")";
                    DatabaseUtilities.update(connection, query);

                            query = "insert into pg_secc_cat values("+id+", "+cat+")";
                    DatabaseUtilities.update(connection, query);

                            //*****************************************************
                            query = "select max(id_texto) from pg_texto";
                    DBResults resultsText = DatabaseUtilities.getQueryResults(connection, query, false);

                            String[] dataText = new String[1];
            dataText = resultsText.getRow(0);
                            if (dataText[0] != null)
                       idText = Integer.parseInt(dataText[0]);
                            else
                              idText = 0;
    idText++;
                            //*****************************************************

                            for (int i=1; i<=va.size(); i++){
                                    query = "insert into pg_texto values("+idText+", NULL, "+id+",
'"+va.get("GNAtexto"+i)+"')";
                                     insert = DatabaseUtilities.update(connection, query);
                            }

                            //*****************************************************
                            query = "select max(id_imagen) from pg_imagen";
                    DBResults resultsImagen = DatabaseUtilities.getQueryResults(connection, query, false);

                            String[] dataImagen = new String[1];
            dataImagen = resultsImagen.getRow(0);
                            if (dataImagen[0] != null)
                       idImagen = Integer.parseInt(dataImagen[0]);
                            else
                              idImagen = 0;
    idImagen++;
                            //*****************************************************

                            //insert images
                            for (int i=1; i<=ve.size(); i++){
                                    String imgTmp = (String)ve.get("GNAimagen"+i);
                                            File imgFile = new File(TmpPath + slash + imgTmp);
                                            String folder = AppPath + "sections" + slash + "sec" + id;
                                            File Folder = new File(folder);
                                            Folder.mkdir();
                                            copyFile(imgFile.toString(), folder);
                                            imgFile.delete();
```

```
                                        query = "insert into pg_imagen values("+idImagen+", NULL, "+id+",
'"+Folder.getName()+"/"+imgTmp+"')";
                                         insert = DatabaseUtilities.update(connection, query);
                                }


                                        //*************************************************
                                        query = "select max(id_file) from pg_file";
                                DBResults resultsFile = DatabaseUtilities.getQueryResults(connection, query, false);

                                        String[] dataFile = new String[1];
                        dataFile = resultsFile.getRow(0);
                                        if (dataFile[0] != null)
                                idFile = Integer.parseInt(dataFile[0]);
                                        else
                                         idFile = 0;
                        idFile++;
                                        //*************************************************

                                //insert files
                                for (int i=1; i<=vi.size(); i++){
                                         String srcTmp = (String)vi.get("GNAfile"+i);
                                                File srcFile = new File(TmpPath + slash + srcTmp);
                                                String folder = AppPath + "sections" + slash + "sec" + id;
                                                File Folder = new File(folder);
                                                Folder.mkdir();
                                                copyFile(srcFile.toString(), folder);
                                                srcFile.delete();

                                        query = "insert into pg_file values("+idFile+", NULL, "+id+",
'"+Folder.getName()+"/"+srcTmp+"')";
                                         insert = DatabaseUtilities.update(connection, query);
                                }

                                getServletConfig().getServletContext().setAttribute("system.updated", "1");
                        }catch(Exception e){
                                handleException(e, "Error, could not save section's data in database");
                                str = "<INPUT TYPE=HIDDEN NAME=accion VALUE=home>\n";
                                str += "<TR><TD COLSPAN=4>Error, could not save section in database. View log for
details.\n";
                                str += "<TR HEIGHT=100><TD COLSPAN=4 ALIGN=CENTER
VALIGN=BOTTOM><INPUT TYPE=SUBMIT VALUE=\"Continue\" CLASS=button>\n";
                                error = true;
                        }finally{
                                try{
                connectionPool.free(connection);
                                }catch(SQLException sql){
                                        handleException(sql, "Error, could not close connection");
                                }

                         if (error){
                                va.put("CONTENT", str);
                                va.put("TITLE", "Section");
                                 template.set_vars(va);
                                String parse = template.print_template("template_seccion.html");
                                        if (parse == null)
                                        out.println(template.last_error);
                                 else
                                out.println(parse);
                                out.flush();
                                out.close();
                                }else{
```

```
                    add_Sec(res);
                }
            }
        }
```

Figura 29. Código para agregar Secciones en Base de Datos.

Podemos notar que dentro este código se agregan imágenes, el texto y en su caso los archivos correspondientes a cada una de las secciones, para distinguirlo el código se encuentra en color azul.

Este método al igual que todos los métodos del sistema para agregar un nuevo componente toma los datos de la pantalla que contiene los campos mínimos necesarios para agregar una sección, esta pantalla es:

Figura 30. Pantalla del BackEnd para agregar una nueva Sección.

El código encargado de agregar una nueva categoría en la base de datos es el siguiente:

```
private void addCatToDB(HttpServletRequest req, HttpServletResponse res)throws ServletException, IOException {
        GnaSvtTemplate template = new GnaSvtTemplate(path);
                Map va = new HashMap();
                String str="", query;
                PrintWriter out = res.getWriter();
                boolean error = false, insert;

                String name = req.getParameter("name");
                String sec  = req.getParameter("sec");

  try {
   Connection connection = connectionPool.getConnection();
                query = "select max(id_cat) from pg_cat";
                DBResults results = DatabaseUtilities.getQueryResults(connection, query, false);

                        int id = 0;
                        String[] data = new String[1];
                        data = results.getRow(0);
                        if (data[0] != null)
                                id = Integer.parseInt(data[0]);
                        id++;

                query = "select nivel from pg_seccion where id_seccion="+sec;
                results = DatabaseUtilities.getQueryResults(connection, query, false);

                        String nivel = "";
                        data = new String[1];
                        data = results.getRow(0);
                        nivel = data[0];

                        query = "insert into pg_cat values("+id+", "+nivel+", '"+name+"')";
                        insert = DatabaseUtilities.update(connection, query);

                        query = "insert into pg_secc_cat values("+sec+", "+id+")";
                        insert = DatabaseUtilities.update(connection, query);

    connectionPool.free(connection);
                }catch(Exception e){
                  handleException(e, "Error, could not save category in database.");
                        str = "<INPUT TYPE=HIDDEN NAME=accion VALUE=home>\n";
                        str += "<TR><TD COLSPAN=4>Error, could not save category in database. View log for
details.\n";
                        str     +=      "<TR    HEIGHT=100><TD    COLSPAN=4    ALIGN=CENTER
VALIGN=BOTTOM><INPUT TYPE=SUBMIT VALUE=\"Continue\">\n";
                        error = true;
                }finally{
                  if (error){
                        va.put("CONTENT", str);
                        va.put("TITLE", "Category");
                          template.set_vars(va);
                        String parse = template.print_template("template_seccion.html");
                                if (parse == null)
                                out.println(template.last_error);
                          else
```

```
                    out.println(parse);
                    out.flush();
                    out.close();
                    }else{
                      handle_Cat(sec, res);
                    }
            }
}
```

Figura 31. Código para agregar Categorías en Base de Datos.

El proceso para modificar una sección o una categoría es muy similar al proceso de insertar, pero cuando lo que se modifica es el template a usar por una sección este template es desechado y se reemplaza por el nuevo.

IV.1.4. Artículos.

Para la administración de los artículos de información se utiliza la clase GnaSvtPortalArt, la cual al igual que las clases anteriormente mencionada cuenta con las acciones de agregar, eliminar y modificar un artículo. Este servlet utiliza las siguientes tablas de la base de datos:

- Pg_articulo. Para agregar información correspondiente al artículo.
- Pg_texto. Para agregar el texto correspondiente a un artículo.
- Pg_imagen. Para agregar las imágenes correspondientes con un artículo.
- Pg_sección. Para asignar un artículo a un a sección.
- Pg_idioma. Para establecer el idota en que estará el artículo.
- Pg_template. Para tomar el template correspondiente al artículo.
- Pg_file. Para agregar archivos necesarios de un template.

Este servlet tiene el siguiente diagrama de clase:

Figura 32. Diagrama de clase para GnaSvtPortalArt.

Utilizando el mismo proceso para agregar un nuevo elemento, en este caso un artículo, lo primero se muestra es la pantalla que confíen la forma de datos necesarios para agregar un nuevo elemento, la cual es generada por el método llamado add_Art(), el cual utiliza un template y escribe los campos de texto que aparecen en la siguiente pantalla:

Figura 33. Pantalla del Back End para agregar un nuevo Template.

Una vez que todos los campos son contienen la información correcta, estos datos son tomados por el método addTodb_Art() para agregarlos dentro de la base de datos, el código que realiza lo anterior es el siguiente:

```
protected void addToDB_Art(HttpServletRequest req, HttpServletResponse res)throws ServletException, IOException {
                String str="", query, title, sec, tpl, exp, summary="", language, dispacher = null;
                int ticker=0;
                Map va = new HashMap();
                Map ve = new HashMap();
                Map vi = new HashMap();
                int count = 0, id;
                PrintWriter out = res.getWriter();
                boolean insert;

    Enumeration params = req.getParameterNames();
    while (params.hasMoreElements()) {
      String name = (String)params.nextElement();
      String value = req.getParameter(name);
                                if (name.startsWith("GNAtexto")){
                                        value = utilz.replaceAll(value, "'", "&#39;");
                                        value = utilz.replaceAll(value, "\"", "&#34;");
                                 va.put(name, value);
                                }else if (name.startsWith("GNAimagen")){
                                        File imgFile = new File(TmpPath + slash + value);
                                 ve.put(name, value);
                                }else if (name.startsWith("GNAfile")){
                                        File srcFile = new File(TmpPath + slash + value);
```

```
                                        vi.put(name, value);
                                    }
    }

                    sec = req.getParameter("GNAsec");
                    tpl = req.getParameter("GNAtpl");
                    language = req.getParameter("language");
                    title = req.getParameter("GNAtitle");
                    summary = req.getParameter("summary");
                    dispacher = req.getParameter("dispacher");

                    if (dispacher == null)
                            dispacher = "";
                    if (summary == null)
                            summary = "";

                    title = utilz.replaceAll(title, "\"", "&#34;");
                    summary = utilz.replaceAll(summary, "\"", "&#34;");

                    if (title==null || sec==null || language==null){
                      out.println("Error, el articulo no existe");
                            out.close();
                    }
                    else{
                            Connection connection = null;
    try {
      connection = connectionPool.getConnection();
                            //insert article
                            query = "select max(id_articulo) from pg_articulo";
                         DBResults results = DatabaseUtilities.getQueryResults(connection,query, false);
            String[] data = new String[1];
               data = results.getRow(0);
                                    if (data[0] != null)
                                id = Integer.parseInt(data[0]);
                                    else
                                      id = 0;
                            id++;

                            query = "insert into pg_articulo values ("+id+", "+sec+", "+tpl+", '"+language+"',
'"+title+"', '"+summary+"', NULL, NULL, '"+dispacher+"')";
                              insert = DatabaseUtilities.update(connection, query);

                            //insert text
                            for (int i=1; i<=va.size(); i++){
                                    query = "insert into pg_texto values("+id+", NULL,
'"+va.get("GNAtexto"+i)+"')";
                                      insert = DatabaseUtilities.update(connection, query);
                            }

                            //insert images
                            for (int i=1; i<=ve.size(); i++){
                                     String imgTmp = (String)ve.get("GNAimagen"+i);
                                            File imgFile = new File(TmpPath + slash + imgTmp);
                                            String folder = AppPath + "articles" + slash + "art" + id;
                                            File Folder = new File(folder);
                                            Folder.mkdir();
                                            copyFile(imgFile.toString(), folder);
                                            imgFile.delete();

                                        query = "insert into pg_imagen values("+id+", NULL,
'"+Folder.getName()+"/"+imgTmp+"')";
                                      insert = DatabaseUtilities.update(connection, query);
```

```
                    }

                    //insert files
                    for (int i=1; i<=vi.size(); i++){
                        String imgTmp = (String)vi.get("GNAfile"+i);
                            File imgFile = new File(TmpPath + slash + imgTmp);
                            String folder = AppPath + "articles" + slash + "art" + id;
                            File Folder = new File(folder);
                            Folder.mkdir();
                            copyFile(imgFile.toString(), folder);
                            imgFile.delete();

                        query = "insert into pg_file values("+id+", NULL,
'"+Folder.getName()+"/"+imgTmp+"')";
                         insert = DatabaseUtilities.update(connection, query);
                    }

                        getServletConfig().getServletContext().setAttribute("system.updated", "1");
                    }catch(Exception e){
                     out.println("Error, could not add article to database. "+e.toString());
                        handleException(e, "Error, could not add article to database.");
                    }finally{
                        try{
            connectionPool.free(connection);
                        }catch(SQLException sql){
                            handleException(sql, "Error, could not close connection");
                        }
                        display_Art(res);
                    }
                }
        }
```

Figura 34. Código para agregar Artículos en Base de Datos.

Como podemos notar en este método para agregar un artículo, antes de agregar el artículo a la base de datos, se abre el template asignado al artículo, y se despliega en pantalla para que todas aquellos campos de texto e imágenes que necesita el template sean insertados; en el caso de los campos de texto, se utiliza otro método para agregar el texto del artículo, pero que también se encarga de reemplazar todos los símbolos como por ejemplo “, ‘,) y demás por el unicode que le corresponde a cada uno, ya que si esto no se hiciera, la momento de desplegar el Artículo en el Front End causaría varios errores. El código correspondiente para agregar texto en un artículo es el que se muestra a continuación:

```
protected void addText_Art(String sec, String tpl, String language, String title, String summary, String exp, String ticker,
String dispacher,   HttpServletResponse res)throws ServletException, IOException {
            GnaSvtTemplate template = new GnaSvtTemplate(TplPath);
                        Map va = new HashMap();
                        PrintWriter out = res.getWriter();
                        String str="", query, screen="",file="", tmp1, tmp2, txt="";
                        int id =0, cmp=0, img=0, n=0, m=0, p=0;
                        Vector vars = new Vector();
                        Vector cont = new Vector();

   try {
     Connection connection = connectionPool.getConnection();
                        query = "select no_campos, no_imagenes, screenshot, archivo from pg_template where
id_template="+tpl;
                        DBResults results = DatabaseUtilities.getQueryResults(connection,query, false);
                          int rows = results.getRowCount();

                        for (int i=0; i<rows; i++){
                          String[] data = new String[4];
                            data = results.getRow(i);
                                    cmp = Integer.parseInt(data[0]);
                                                img = Integer.parseInt(data[1]);
                                                screen = data[2];
                                                file = data[3];
                        }

                                    char Slash = slash.charAt(0);
                                    file = file.replace('/', Slash);

                        query = "select nombre from pg_variable where (nombre like ('%_TXT') or nombre like ('%_IMG') or
nombre like ('%_FILE')) and id_template="+tpl;
                        results = DatabaseUtilities.getQueryResults(connection,query, false);
                          rows = results.getRowCount();

                        for (int i=0; i<rows; i++){
                          String[] data = new String[1];
                            data = results.getRow(i);
                                                vars.add(data[0]);
                        }

                                    title = utilz.replaceAll(title, "\"", "&#34;");
                                    summary = utilz.replaceAll(summary, "\"", "&#34;");

                        str += "<SCRIPT LANGUAGE=\"JavaScript\"
src=\""+Virtual+"admin/insertText_templates.js\"></script>\n";
                        str += "<FORM NAME=\"GNAFORMA\"
ACTION=\""+Servlet+"mx.com.gedas.portal.GnaSvtPortalArt\" METHOD=POST>\n";
                          str += "<INPUT TYPE=HIDDEN NAME=accion VALUE=\"addToDB\">\n";
                                    str += "<INPUT TYPE=HIDDEN NAME=GNAsec VALUE=\""+sec+"\">\n";
                        str += "<INPUT TYPE=HIDDEN NAME=GNAtpl VALUE=\""+tpl+"\">\n";
                                    str += "<INPUT TYPE=HIDDEN NAME=language VALUE=\""+language+"\">\n";
                        str += "<INPUT TYPE=HIDDEN NAME=GNAtitle VALUE=\""+title+"\">\n";
                                    if (!summary.equals("")) str += "<INPUT TYPE=HIDDEN NAME=summary
VALUE=\""+summary+"\">\n";
                                    if (!dispacher.equals("")) str += "<INPUT TYPE=HIDDEN NAME=dispacher
VALUE=\""+dispacher+"\">\n";
                                    str += "<INPUT TYPE=HIDDEN NAME=Virtual VALUE=\""+Virtual+"\">\n";

                                    String js = "<SCRIPT>\n";
                        for (int i=0; i<vars.size(); i++){
```

```
                    txt = "";
                     if(vars.elementAt(i).toString().endsWith("_TXT")){
                                m++;
                                txt += "<INPUT TYPE=BUTTON NAME=GNAtexto"+m+" VALUE=\"Insertar Texto
"+m+"\" onclick=\"OpenDialog('txt', 'GNAtexto"+m+"')\"><BR>\n";
                                    str += "<INPUT TYPE=HIDDEN NAME=GNAtexto"+m+">\n";
                                        js += "var GNAtexto"+m+" = \"\";\n";
                                }else if(vars.elementAt(i).toString().endsWith("_IMG")){
                                 n++;
                                  txt += "<INPUT TYPE=BUTTON NAME=GNAimagen"+n+" VALUE=\"Insertar
Imagen " +n+"\" onclick=\"OpenDialog('img', 'GNAimagen"+n+"')\"><BR>\n";
                                    str += "<INPUT TYPE=HIDDEN NAME=GNAimagen"+n+">\n";
                                        js += "var GNAimagen"+n+" = \"\";\n";
                                }else if(vars.elementAt(i).toString().endsWith("_FILE")){
                                 p++;
                                  txt += "<INPUT TYPE=BUTTON NAME=GNAfile"+p+" VALUE=\"Insertar Archivo
"+p+"\" onclick=\"OpenDialog('file', 'GNAfile"+p+"')\"><BR>\n";
                                    str += "<INPUT TYPE=HIDDEN NAME=GNAfile"+p+">\n";
                                        js += "var GNAfile"+p+" = \"\";\n";
                                }
                                 va.put(vars.elementAt(i).toString(), txt);
                    }

                    str += "</FORM>\n";
                                js  += "</SCRIPT>\n";
                                str += js;
          va.put("BEGIN_BODY", str);
          va.put("END_BODY", "<INPUT TYPE=BUTTON VALUE=\"Aceptar\" onClick=\"GNACheck()\">");
                                connectionPool.free(connection);
                    }
                    catch(Exception e){
          va.put("BEGIN_BODY", "Error, no se pudo mostrar el template. "+e.toString());
                                log.write(host, e.toString());
                    }finally{
                     String tplPath = file.substring(0, file.indexOf(slash));
        va.put("TITLE", title);
           template.set_vars(va);
           String parse = template.print_template(file, tplPath);
                    if (parse == null)
          out.println(template.last_error);
           else
             out.println(parse);
        out.flush();
           out.close();
                    }
 }
```

Figura 35. Código para agregar el texto correspondiente a un Artículo en Base de Datos.

En lo que respecta a modificar un artículo el proceso también es muy similar al proceso de de insertar, ya que se toman los datos correspondientes a ese articulo y al ser modificados, se modifican en la base de datos.

IV.1.5. Noticias.

Para terminar con la parte del Back End debemos explicar el Servlet encargado de las noticias, el cual además de tener las funciones básicas de agregar, eliminar y modificar, contiene una opción para desplegar dentro de un template, esto siempre y cuando el template cuente con la funcionalidad necesaria para soportarlo.

Las tablas que utiliza este Servlet son:

- Pg_articulo. Para agregar la información correspondiente a la noticia, es importante mencionar que la diferencia entre un artículo y una noticia la hace el campo llamado “expira”, el cual si su valor es 0, significa que es un artículo, y si es 1 significa que es una noticia.
- Pg_texto. Para guardar el texto correspondiente a una noticia.

Este servlet tiene el siguiente diagrama de clase:

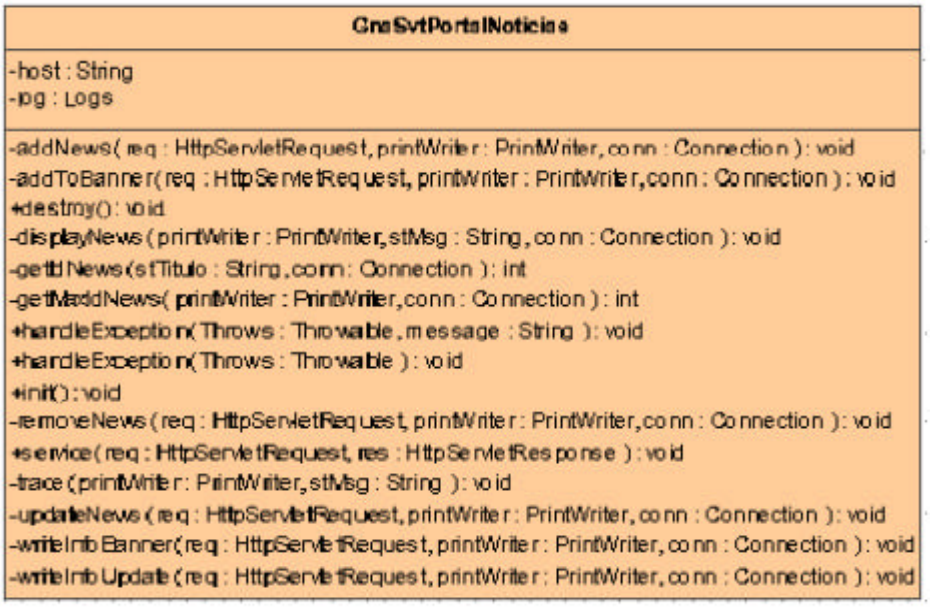

Figura 36. Diagrama de clase para GnaSvtPortalNoticias.

Como podemos ver los métodos más importantes son el de agregar a base de datos (addNews()), el cual trabaja de igual manera que para agregar un artículo, pero con la diferencia todas las noticias tendrán el valor de 1 dentro de la tabla pg_articulo, en el campo expira. Como lo podemos notar en el siguiente código:

```
private void addNews(HttpServletRequest req, PrintWriter printWriter, Connection conn)
  {
    try{
      int inMaxIdNews = getMaxIdNews(printWriter, conn);
      String stQuery = "insert into pg_articulo(id_articulo, id_seccion, id_template, id_idioma, titulo, resumen, expira,
ticker, dispacher)";
      stQuery += " values(" + (inMaxIdNews + 1) + ", null, null, " + req.getParameter("idioma") + ", '" +
req.getParameter("tituloNews") + "', null, null, 1, null)";
      Statement stNews = conn.createStatement();
      stNews.executeUpdate(stQuery);

      stQuery = "insert into pg_texto(id_articulo, texto)";
      stQuery += " values(" + (inMaxIdNews + 1) + ", '" + req.getParameter("dsNews") + "')";
      stNews.executeUpdate(stQuery);

      displayNews(printWriter, "Noticia dada de alta satisfactoriamente.", conn);
      conn.close();
    }
    catch(SQLException sqlexc){
      trace(printWriter, sqlexc.toString());
      displayNews(printWriter, "No pudo darse de alta la noticia satisfactoriamente.", conn);
    }
  }
```

Figura 37. Código para agregar Noticias en Base de Datos.

El método utilizado de agregar una noticia al ticker de un template consta de modificar el valor del campo ticker dentro de la tabla pg_articulo, ya que si este valor es 1, quiere decir que la noticia será mostrada en el Front End, mientras que en caso contrario no se mostrará en el FrontEnd. Esto lo podemos notar a detalle en el siguiente código:

```
private void addToBanner(HttpServletRequest req, PrintWriter printWriter, Connection conn)
  {
    try{
      String stTempBanner = "";
```

```
    String stQuery0 = "update pg_articulo set ticker=0 where id_articulo in (";
    String stQuery1 = "update pg_articulo set ticker=1 where id_articulo in (";
    Statement stNews = conn.createStatement();
    // Query 1
    StringTokenizer sttOrigen = new StringTokenizer(req.getParameter("tituloNews"), "*");
    if(sttOrigen.countTokens() > 0)
    {
      stTempBanner = sttOrigen.nextToken();
      stQuery0 += stTempBanner.substring(0, stTempBanner.indexOf("|"));

      while(sttOrigen.hasMoreTokens())
      {
        stTempBanner = sttOrigen.nextToken();
        stQuery0 += ", " + stTempBanner.substring(0, stTempBanner.indexOf("|"));
      }
      stQuery0 += ")";
      //trace(printWriter, "Query0: " + stQuery0);
      stNews.executeUpdate(stQuery0);
    }

    // Query 2
    StringTokenizer sttDestino = new StringTokenizer(req.getParameter("dsNews"), "*");
    if(sttDestino.countTokens() > 0)
    {
      stTempBanner = sttDestino.nextToken();
      stQuery1 += stTempBanner.substring(0, stTempBanner.indexOf("|"));

      while(sttDestino.hasMoreTokens())
      {
        stTempBanner = sttDestino.nextToken();
        stQuery1 += ", " + stTempBanner.substring(0, stTempBanner.indexOf("|"));
      }
      stQuery1 += ")";
      //trace(printWriter, "Query1: " + stQuery1);
      stNews.executeUpdate(stQuery1);
    }

    displayNews(printWriter, "Proceso realizado satisfactoriamente.", conn);
    conn.close();
  }
  catch(SQLException sqlexc){
    trace(printWriter, sqlexc.toString());
    displayNews(printWriter, "Proceso no realizado satisfactoriamente.", conn);
  }
}
```

Figura 38. Código para agregar una Noticia a un Banner en el Fornt End.

Como se menciono anteriormente dentro de este capitulo, la librería de O´Reilly [Java O´Reilly] será utilizada en el BackEnd, en la parte de Templates, ya que un template puede ser tomado desde cualquier computadora en la que se encuentre el usuario y es

necesario que al agregar el template a la aplicación este sea colocado en la carpeta que se llama Templates.

Ahora bien en lo referente al Controller del Back End tenemos que mencionar que cada uno de los servlets anteriormente explicados cuenta con un método llamado service que es quien se encarga de controlar el despliegue de información, es decir de acuerdo a la petición que s ele hace al servlets este ejecutará el método correspondiente a la petición hecha.

El método service para el Servlet de los templates es el siguiente:

```
public void service(HttpServletRequest req, HttpServletResponse res)throws ServletException, IOException {
                    PrintWriter out = res.getWriter();
  res.setContentType("text/html");
  String cmd, val, menu;

                    try{

    session = validateSession (req);
                    cmd = (String)req.getParameter("accion");
                     host = req.getRemoteHost();
                              user = validateUser(session);

                    if (cmd != null){
                     if (cmd.equals("menu") && session != null){
                               display_Tpl(res);
                               }else if (cmd.equals("home") && session != null){
                                 display_Tpl(res);
                              }else if (cmd.equals("addTpl") && session != null){
                                        addTpl(req, res);
                              }else if (cmd.equals("addToDB") && session != null){
                                        addToDB(req, res);
                              }else if (cmd.equals("variablestoBD") && session != null){
                                        variablestoBD(req, res);
                              }else if (cmd.equals("deleteTpl") && session != null){
                                        deleteTpl(req, res, req.getParameter("tpl"));
                              }else if (cmd.equals("updateTpl") && session != null){
                                        modifyTpl(req.getParameter("tpl"), res);
                              }else if (cmd.equals("updateDB") && session != null){
                                        modifyDB(req, res);
                              }else if (cmd.equals("openHTML") && session != null){
                                        parserHTML(req, res);
                              }else if (cmd.equals("openScreen") && session != null){
                                        parserScreen(req, res);
                              }else if (cmd.equals("updatetoDB") && session != null){
                                        updateToDB(req, res);
                              }else if (cmd.equals("modifytoDB") && session != null){
```

```
                    modifyToDB(req, res);
                }
            else
                    if (session != null)
                out.println("Error, missing arguments");
                        else
                          res.sendRedirect (defaultPage);
            }
            else
res.sendRedirect(defaultPage);
}catch (Exception e){
             handleException(e);
            }
}
```

Figura 39. Código para el método Service de un Servlet.

Aquí podemos ver que la petición siempre llega por medio de la variable llamada cmd, y que en base a esta se ejecutará el método que satisface a esa petición, por ejemplo:

```
if (cmd.equals("addToDB") && session != null){

    addToDB(req, res);
```

Si la petición es “addtoDB” centremos que ejecutar el método addToDB(req, res), el cual es el encargado de almacenar la información dentro de la base de datos.

Este patrón se usa dentro de los demás servlets, ya que cada uno de ellos tiene un método service donde recibe las peticiones por parte del usuario y las resuelve de acuerdo a los métodos con los que cuenta el servlet.

Hablando de la tercera y última capa de la tecnología “Model View Controller”, tenemos que explicar que la capa de presentación View, esta dentro de cada uno de los servlets de secciones, artículos, templates y noticias, ya que cada uno de ellos cuenta con

métodos para desplegar la pantallas con la información necesaria para agregar, eliminar o modificar la información correspondiente. Un ejemplo de ello es el siguiente método del Servlet de Templates, el cual se encarga de mostrar la forma con los datos necesarios para agregar un template, el código es el siguiente:

```
protected void addTpl(HttpServletRequest req, HttpServletResponse res)throws ServletException, IOException {
        GnaSvtTemplate template = new GnaSvtTemplate(path);
        Map va = new HashMap();
        String txt = "", query;
        PrintWriter out = res.getWriter();

                txt += "<INPUT TYPE=\"hidden\" NAME=\"accion\" VALUE=\"addToDB\">\n";
                txt += "<tr>\n";
    txt += "  <td class=\"content\">Name</td>\n";
    txt += "  <td colspan=3><input name=\"nombre\" type=\"text\" size=38 CLASS=\"field\"></td>\n";
    txt += "</tr>\n";

                txt += "<tr>\n";
    txt += "  <td class=\"content\">ZIP File</td>\n";
    txt += "  <td  colspan=3><input name=\"zip\" type=\"text\" size=30 CLASS=\"field\" > <input
TYPE=\"button\" VALUE=\"Choose\" NAME=\"img\" onClick=\"agregarHTMLAgregar();\"
CLASS=\"button\"></td>\n";
    txt += "</tr>    \n";

    txt += "<tr>\n";
    txt += "  <td class=content>Screenshot</td>\n";
    txt += "  <td  colspan=3><input name=\"screenshot\" type=\"text\" size=30 CLASS=\"field\" > <input
TYPE=\"button\" VALUE=\"Choose\" NAME=\"img\" onClick=\"agregarIMGScreenshot();\"
CLASS=\"button\"></td>\n";
    txt += "</tr>    \n";

    txt += "<tr>\n";
    txt += "  <td class=content>Type</td>\n";
    txt += "  <td  colspan=3><input name=\"kind\" type=\"radio\" VALUE=\"sec\"> Section\n";
    txt += "              <input name=\"kind\" type=\"radio\" VALUE=\"art\"> Article</td>\n";
    txt += "</tr>    \n";

                txt += "<tr>\n";
                txt += "  <td colspan=4><table width=100%>\n<tr>\n";
                txt += "   <td align=\"center\" height=\"50\"><INPUT TYPE=\"BUTTON\"
VALUE=\"Accept\" CLASS=\"button\" onClick=\" check_forma();\"></td>\n";
                txt += "   <td align=\"center\" height=\"50\"><INPUT TYPE=\"BUTTON\"
VALUE=\"Cancel\" CLASS=\"button\" onClick=\" command('continue', false)\"></td>\n";
                txt += "</tr></table></td>\n";
                txt += "</tr>\n";

                txt += "<tr>\n";
    txt += "  <td colspan=\"4\" height=\"50\" valign=\"bottom\"><FONT SIZE=2><B>Note:</B> Screenshot's
dimensions must be 300x200 pixels.</FONT></td>\n";
    txt += "</tr>\n";

        va.put("CONTENT", txt);
        va.put("TITLE", "New template");
          template.set_vars(va);
```

```
String parse = template.print_template("template_template.html");
            if (parse == null)
out.println(template.last_error);
else
  out.println(parse);

out.flush();
 out.close();
}
```

Figura 40. Código para agregar un Templates en Base de Datos.

Aquí notamos que este método hace uso del template HTML llamado "template_templates.html", el cual se lee y al momento de encontrar la variable CONTENT dentro del template se reemplazara por la variable TXT del servlet, lo cual dará como resultado la siguiente pantalla:

Figura 41. Pantalla del BackEnd para agregar un nuevo template.

Así como trabaja este método trabajan muchos otros dentro lo servlets antes mencionados, donde se usa un template especifico con una variable y esta variable es reemplazada por el código correspondiente de un método, lo cual resulta en una página

HTML como la mostrada anteriormente. El propósito de esto es reducir el número de páginas HTML para cada opción del sistema y así únicamente tener un template por opción del sistema y no por método para cada acción de una opción del sistema.

IV.2. Front End.

Ahora tenemos que explicar todo lo referente al Front End, y para ello empezaremos por explicar lo referente a la lógica de manejo de Información (Model). Primero tenemos que mencionar que los archivos desarrollados para esta capa fueron los siguientes:

Model para Front End

Nombre	**Descripción**
GnaBeansFront	Clase que proporciona la interfaz principal del Front End por medio de un archivo JSP y la base de datos.
GnaBeansObjArticle	Clase encargada de desplegar los artículos de información del sitio, añadiéndoles la funcionalidad correspondiente a cada artículo.
GnaBeansObjSection	Clase encargada de desplegar las secciones de información del sitio, añadiéndoles la funcionalidad correspondiente a cada sección.

Tabla 3. Código Descripción de módulos para Front End.

IV.2.1. Model.

Empecemos por la clase más importante dentro del Front, GnaSvtBeansfront, ya que confine todas las variables y directivas necesarias para darle al index, a cada sección y artículo la funcionalidad correspondiente, es decir, que crea las ligas entre los Articulos y secciones y además genera el path para cada imagen utilizada dentro del sitio.

Todo esto podemos verlo a detalle en el método init() de la clase GnaBeansFront, pero antes veamos el diagrama de clase.

Figura 42. Diagrama de clase para GnaBeansFront.

```
public void init(String file, HttpSession Session, ServletContext context) {

        String var;
  int vendor = DriverUtilities.MSSQL;
  String driver = DriverUtilities.getDriver(vendor);
                session = Session;
                Context = context;

                GnaSvtTemplate.File = file;
                GnaParser conf = new GnaParser(file);
                Map params      = conf.getInitParams();
                path                            = (String)params.get("aplicationpath") + "admin" + slash;
                tipsFile        = (String)params.get("aplicationsetup") + "messages.cfg";
                var                             = (String)params.get("aplicationsetup") + "servletPortal.cfg";
                newsletter      = (String)params.get("aplicationsetup") + "newsletter.dat";
                defaultPage = (String)params.get("defaultpage");
                AppPath                 = (String)params.get("virtualpath");
                Servlet                 = (String)params.get("servletpath");
                TplPath                 = (String)params.get("templates");
                ImgPath                 = (String)params.get("imagespath");

                Map vars = DriverUtilities.read_conf_file(var);
  String Host    = vars.get("server").toString().trim();
  String dbName  = vars.get("DBname").toString().trim();
  String username = vars.get("user").toString().trim();
  String password = vars.get("password").toString().trim();
  String url = DriverUtilities.makeURL(Host, dbName, vendor);

        try{
                log = new Logs("C:\\temp", "BCVWM_front.log");
            }catch(Exception e){
              handleException(e);
            }


  try {
                if (connectionPool == null){
                        connectionPool = new ConnectionPool(driver, url, username, password, 10, 50,
true);
                        context.log("[GnaSvtPortalManager Front] Initializing connection pool: "+Host);
                }
  } catch(SQLException sqle) {
                context.log("[GnaSvtPortalManager Front] Error initializing connection pool: " + sqle);
    connectionPool = null;
  }

                SystemVars = new String[16];
        SystemVars[0] = "System.Date";                                          //print current date
          SystemVars[1] = "Section.Name";                                       //print section's name of
current secc_padre
          SystemVars[2] = "Section.Link";                                       //print section's link of
current secc_padre
          SystemVars[3] = "Article.Title";                                  //print article's title of current
section
          SystemVars[4] = "Article.Summary";                        //print article's summary
          SystemVars[5] = "Article.Link";                                       //print article's link of
current section
                SystemVars[6] = "System.Counter";                               //print counter's actual
value
                SystemVars[7] = "getSectionLink";                               //print section's link
from a specific section's name. ie: String getSectionLink(seccion)
```

```
                SystemVars[8] = "getChildSections";                    //print a section child list from a
specific section. ie: String[] getChildSections(seccion)
                SystemVars[9] = "getArticleLink";                              //print a article's link
from a specific article's name. ie: String getArticleLink(article)
                SystemVars[10] = "System.Home";                                //implemets
only for BCVWM
                SystemVars[11] = "System.Sitemap";                     //implemets only for BCVWM
                SystemVars[12] = "Section.Parent";                     //print parent of current section
                SystemVars[13] = "Article.Parent";                     //print parent section of current
article
                SystemVars[14] = "System.Dispacher";                   //include a JSP file from current
dispacher
                SystemVars[15] = "System.Home.Swap";                   //implemets only for BCVWM for
Change language

                String updated = (String) context.getAttribute("system.updated");

                //Cache inizialitation
                if (user_request == null)
                        user_request = new String[100];
                else if (updated != null && updated.equals("1")){
                        user_request = new String[100];
                        context.setAttribute("system.updated", "0");
                }

                if (request_data == null)
                        request_data = new Vector(100);
                else if (updated != null && updated.equals("1")){
                        request_data = new Vector(100);
                        context.setAttribute("system.updated", "0");
                }
 }
```

Figura 43. Código para inicializar parámetros de la aplicación.

Podemos apreciar que además de generar las ligas dentro del sistema y reemplazar las directivas por el código correspondiente, esta clase es la encargada de intercambiar el idioma del sitio, siempre y cuando el sitio este programado para ello.

Otro método muy importante dentro de esta clase, es le método getMainSections(), ya que se encarga de traer las secciones principal del sitio para así desplegarlas dentro del sitio, vemos a detalle este método:

```
public String[][] getMainSections(String category, String Language){
  String[][] retValues = null;
```

```
                String query;

                String cache = "getMainSections:" + category + ";" + Language;
                if ((user_request != null) && (cacheContains(cache))){
                  writeLog(connectionPool.toString());
                        return ( (String[][])getCacheData(cache) );
                }else{

                Connection connection = null;
                Exception ex = new Exception("Error");
   try {
    connection = connectionPool.getConnection();
                        handleException(ex, "getMainSections("+category+", "+Language+") -
"+connectionPool.toString());


                        query = "select s.id_seccion, s.descripcion from pg_seccion s, pg_secc_cat sc, pg_cat c
with(nolock) where  s.secc_padre is null and c.id_cat = sc.id_cat and s.id_seccion=sc.id_seccion and s.nivel <> c.nivel
and s.id_idioma=? and c.descripcion like (?) order by s.id_seccion";
                        String[] params = {Language, category};
                        DBResults results = DatabaseUtilities.getQueryResults(connection, query, params, true);
//connection closed

                  int rows = results.getRowCount();
                        retValues = new String[rows][2];

                for (int i=0; i<rows; i++){
                  //String[] data = new String[2];
                                String[] data;
                    data = results.getRow(i);
                                retValues[i][0] = data[0];
                                retValues[i][1] = data[1];
                }

   } catch(Exception e) {
            //trace("Error: "+e.toString());
                        handleException(e, "getMainSections("+category+", "+Language+") - Error -
"+e.toString());
   }finally{
                        try{
                connectionPool.free(connection);
                                handleException(ex, "getMainSections("+category+", "+Language+") - closed -
"+connectionPool.toString());
                        }catch(SQLException sql){
                                writeLog("No se pudeo cerrar conexion en: getMainSections("+category+",
"+Language+")");
                        }
                }
                setCacheData(cache, retValues);
                }
                return retValues;
        }
```

Figura 44. Código para obtener las secciones principales de un sitio.

Otro método muy importante es getSections() donde se traen todas y cada una de las secciones del sitio, pero estas secciones son traídas en forma de objeto, ya que cuando una de ellas se pedida por el usuario se desplegará lista para navegarse, lo que quiere decir que las ligas estarán activas, los botones ejecutarán las acciones correspondientes. El código que realiza todo lo anterior es le siguiente:

```
public String getSection(String idSec, String idLanguage){
  GnaSvtTemplate template = new GnaSvtTemplate(TplPath);
        Map va = new HashMap();
        String templateFile="", query;
        StringBuffer parse = new StringBuffer();
        int tpl;
        Vector section_id = new Vector();
        Vector section_parent = new Vector();
        Vector section_language = new Vector();
        Vector section_name = new Vector();
        Vector section_idCat = new Vector();
        Vector section_nameCat = new Vector();
        Vector article_id = new Vector();
        Vector article_section = new Vector();
        Vector article_title = new Vector();
        Vector article_summary = new Vector();
        Vector article_dispacher = new Vector();
        Vector article_language = new Vector();
        Vector article_texto = new Vector();
        Vector article_imagen = new Vector();
        Vector article_file = new Vector();
        long init_time = System.currentTimeMillis();

        String cache = "getSection:" + idSec + ";" + idLanguage;
        if ((user_request != null) && (cacheContains(cache))){
          writeLog(connectionPool.toString());
              return ( (String)getCacheData(cache) );
        }else{

        Connection connection = null;
        Exception ex = new Exception("Error");
try {
  connection = connectionPool.getConnection();
              handleException(ex, "getSection(idSec, idLanguage) - "+connectionPool.toString());
        //get id template
        //query = "select id_template from pg_seccion where id_seccion="+idSec;
        //query = "select id_template from pg_seccion with(nolock) where id_seccion="+idSec;
        //DBResults results = DatabaseUtilities.getQueryResults(connection,query, false);

        query = "select id_template from pg_seccion with(nolock) where id_seccion=?";
              String[] params = {idSec};
              DBResults results = DatabaseUtilities.getQueryResults(connection, query, params, false);

    String[] data = new String[1];
      data = results.getRow(0);
              tpl = Integer.parseInt(data[0]);
```

```
                    //get template file
                    //query = "select archivo from pg_template where id_template="+tpl;
                    //query = "select archivo from pg_template with(nolock) where id_template="+tpl;
                    //results = DatabaseUtilities.getQueryResults(connection,query, false);

                    query = "select archivo from pg_template with(nolock) where id_template=?";
                              params[0] = ""+tpl;
                              results = DatabaseUtilities.getQueryResults(connection, query, params, false);

          data = new String[1];
           data = results.getRow(0);
                              templateFile = data[0];

                    //get variable data
                    //query = "select nombre, contenido from pg_variable where nombre not like ('%_TXT') and nombre
not like ('%_IMG') and nombre not like ('%_FILE') and id_template="+tpl;
                    //query = "select nombre, contenido from pg_variable with(nolock) where nombre not like ('%_TXT')
and nombre not like ('%_IMG') and nombre not like ('%_FILE') and id_template="+tpl;
                    //results = DatabaseUtilities.getQueryResults(connection,query, false);

                    query = "select nombre, contenido from pg_variable with(nolock) where nombre not like ('%_TXT')
and nombre not like ('%_IMG') and nombre not like ('%_FILE') and id_template=?";
                              params[0] = ""+tpl;
                              results = DatabaseUtilities.getQueryResults(connection, query, params, false);

                              int rows = results.getRowCount();

                              String[][] templateVars = new String[rows][2];
                              for (int i=0; i<rows; i++){
           data = new String[2];
            data = results.getRow(i);
                                        templateVars[i][0] = data[0];
                                        templateVars[i][1] = data[1];
                              }

                    //get user variable data
                    //query = "select nombre from pg_variable where (nombre like ('%_TXT') or nombre like ('%_IMG')
or nombre like ('%_FILE')) and id_template="+tpl;
                    //query = "select nombre from pg_variable with(nolock) where (nombre like ('%_TXT') or nombre like
('%_IMG') or nombre like ('%_FILE')) and id_template="+tpl;
                    //results = DatabaseUtilities.getQueryResults(connection,query, false);

                    query = "select nombre from pg_variable with(nolock) where (nombre like ('%_TXT') or nombre like
('%_IMG') or nombre like ('%_FILE')) and id_template=?";
                              params[0] = ""+tpl;
                              results = DatabaseUtilities.getQueryResults(connection, query, params, false);

                              rows = results.getRowCount();

                              String[] templateUserVars = new String[rows];
                              for (int i=0; i<rows; i++){
           data = new String[1];
            data = results.getRow(i);
                                        templateUserVars[i] = data[0];
                              }

                    //get articles data
                    //query = "select id_articulo, id_seccion, id_idioma, titulo, resumen, dispacher from pg_articulo
where id_seccion is not null and id_idioma="+idLanguage+" order by id_articulo, id_seccion";
                    //query = "select id_articulo, id_seccion, id_idioma, titulo, resumen, dispacher from pg_articulo
with(nolock) where id_seccion is not null and id_idioma="+idLanguage+" order by id_articulo, id_seccion";
```

```
//results = DatabaseUtilities.getQueryResults(connection,query, false);

query = "select id_articulo, id_seccion, id_idioma, titulo, resumen, dispacher from pg_articulo with(nolock) where id_seccion is not null and id_idioma=? order by id_articulo, id_seccion";
params[0] = idLanguage;
results = DatabaseUtilities.getQueryResults(connection, query, params, false);

rows = results.getRowCount();

for (int i=0; i<rows; i++){
data = new String[5];
 data = results.getRow(i);
article_id.add(data[0]);
if (data[1] == null)
article_section.add("NULL");
else
article_section.add(data[1]);
article_language.add(data[2]);
article_title.add(data[3]);
article_summary.add(data[4]);
article_dispacher.add(data[5]);
}

//get sections data
//query = "select s.id_seccion, s.secc_padre, s.id_idioma, s.descripcion, c.id_cat, cat.descripcion from pg_seccion s, pg_secc_cat c, pg_cat cat where s.id_seccion = c.id_seccion and c.id_cat = cat.id_cat and s.id_idioma="+idLanguage+" and s.id_seccion <> 0 order by s.id_seccion";
//query = "select s.id_seccion, s.secc_padre, s.id_idioma, s.descripcion, c.id_cat, cat.descripcion from pg_seccion s, pg_secc_cat c, pg_cat cat with(nolock) where s.id_seccion = c.id_seccion and c.id_cat = cat.id_cat and s.id_idioma="+idLanguage+" and s.id_seccion <> 0 order by s.id_seccion";
//results = DatabaseUtilities.getQueryResults(connection,query, false);

query = "select s.id_seccion, s.secc_padre, s.id_idioma, s.descripcion, c.id_cat, cat.descripcion from pg_seccion s, pg_secc_cat c, pg_cat cat with(nolock) where s.id_seccion = c.id_seccion and c.id_cat = cat.id_cat and s.id_idioma=? and s.id_seccion <> 0 order by s.id_seccion";
params[0] = idLanguage;
results = DatabaseUtilities.getQueryResults(connection, query, params, false);

rows = results.getRowCount();

for (int i=0; i<rows; i++){
data = new String[6];
 data = results.getRow(i);
section_id.add(data[0]);
if (data[1] == null)
section_parent.add("NULL");
else
section_parent.add(data[1]);
section_language.add(data[2]);
section_name.add(data[3]);
section_idCat.add(data[4]);
section_nameCat.add(data[5]);
}

//Main section object
GnaObjSection section = new GnaObjSection(section_id, section_parent, section_language, section_name, section_idCat, section_nameCat);
//Main article object
GnaObjArticle article = new GnaObjArticle(article_id, article_section, article_language, article_title, article_summary, article_dispacher);

////////////////////////////////////
```

```
                    //Get Section texts, images and files
                    //get section's texto
                //query = "select texto from pg_texto where id_seccion="+idSec+" order by id_texto";
                //query = "select texto from pg_texto with(nolock) where id_seccion="+idSec+" order by id_texto";
                //results = DatabaseUtilities.getQueryResults(connection, query, false);

                query = "select texto from pg_texto with(nolock) where id_seccion=? order by id_texto";
                    params[0] = idSec;
                    results = DatabaseUtilities.getQueryResults(connection, query, params, false);

                    rows = results.getRowCount();

                    for (int i=0; i<rows; i++){
        data = new String[1];
          data = results.getRow(i);
                    article_texto.add(data[0]);
                    }

                    //get section's texto
                //query = "select path from pg_imagen where id_seccion="+idSec+" order by id_imagen";
                //query = "select path from pg_imagen with(nolock) where id_seccion="+idSec+" order by
id_imagen";
                //results = DatabaseUtilities.getQueryResults(connection, query, false);

                query = "select path from pg_imagen with(nolock) where id_seccion=? order by id_imagen";
                    params[0] = idSec;
                    results = DatabaseUtilities.getQueryResults(connection, query, params, false);

                    rows = results.getRowCount();

                    for (int i=0; i<rows; i++){
        data = new String[1];
          data = results.getRow(i);
                    article_imagen.add(data[0]);
                    }

                    //get section's texto
                //query = "select path from pg_file where id_seccion="+idSec+" order by id_file";
                //query = "select path from pg_file with(nolock) where id_seccion="+idSec+" order by id_file";
                //results = DatabaseUtilities.getQueryResults(connection, query, false);

                query = "select path from pg_file with(nolock) where id_seccion=? order by id_file";
                    params[0] = idSec;
                    results = DatabaseUtilities.getQueryResults(connection, query, params, true);  //connection
closed

                    rows = results.getRowCount();

                    for (int i=0; i<rows; i++){
        data = new String[1];
          data = results.getRow(i);
                    article_file.add(data[0]);
                    }

                    //get Directives & SystemVars from template's variables
                    Vector[] templateDirectives = new Vector[templateVars.length];
                    Vector[] templateVariables  = new Vector[templateVars.length];
                    for (int i=0; i<templateVars.length; i++){
                        String[] directivesArray = new String[countDirectives(templateVars[i][1])];
                        directivesArray = getDirectives(templateVars[i][1]);

                        Map directives = new HashMap();
```

```
                    for (int j=0; j<directivesArray.length; j++)
                      directives.put(directivesArray[j], getDirectiveValue(templateVars[i][1],
directivesArray[j]));


                    String[] systemVars = new String[countSysVars(templateVars[i][1])];
                    systemVars = getSystemVars(templateVars[i][1]);
                    templateDirectives[i] = new Vector();
                    templateDirectives[i].add(directives);  //Directives list of each template's variable
                    templateVariables[i] = new Vector();
                    templateVariables[i].add(systemVars);    //System variables list of each template's
variable
                    templateVars[i][1]  =  eraseDirectives(templateVars[i][1]);    //delete  directives
from template's variable
                }

                parse.append(templateVars.length + " total<bR>");
                //////////////////////
                //Process template's system variables
                //////////////////////
                for (int i=0; i<templateVars.length; i++){
                    //Init directives
                    int Counter = 0;
                    String Languaje = "ESPAÑOL";
                    String Category = null;
                    String Parent   = null;
                    String Parser   = "TRUE";
                    int LN = GnaObjSection.SPANISH;

                    String VarInProcess = "";
                    boolean processed = false;

                    //Process directives
                    HashMap dirs = (HashMap) templateDirectives[i].elementAt(0);
                    if (dirs.containsKey("LANGUAGE")) Languaje = (String)
dirs.get("LANGUAGE");
                    if (dirs.containsKey("CATEGORY")) Category = (String) dirs.get("CATEGORY");
                    if (dirs.containsKey("PARENT"))   Parent = (String) dirs.get("PARENT");
                    if (dirs.containsKey("COUNTER")) Counter =
Integer.parseInt(dirs.get("COUNTER").toString());
                    if (dirs.containsKey("PARSER"))   Parser = (String) dirs.get("PARSER");

                    if (Languaje.equals("ESPAÑOL"))
                        LN = GnaObjSection.SPANISH;
                    else if (Languaje.equals("INGLES"))
                        LN = GnaObjSection.ENGLISH;
                    //LN = idLanguage;

                    if (Parent == null)
                        Parent = idSec;
                    else if (Parent.equalsIgnoreCase("Home"))
                      Parent = "NULL";
                    else if (Parent.equalsIgnoreCase("{"+ SystemVars[12] +"}")){
                        String padre = section.getParent(idSec);
                        Parent = padre;
                    }else if (Parent.equalsIgnoreCase("{"+ SystemVars[13] +"}")){
                        //String padre = section.getParent(idSec);
                        Parent = idSec;
                    }else
                        Parent = section.getID(Parent, idLanguage);

                    //Get System variables from template's variable
```

```
String[] tmpSysVars = (String[]) templateVariables[i].elementAt(0);
int count=0,pos=0;
for (int j=0; j<tmpSysVars.length; j++)
        if (tmpSysVars[j] != null)
          count++;

String[] SysVars = new String[count];
for (int j=0; j<tmpSysVars.length; j++)
        if (tmpSysVars[j] != null)
                SysVars[pos++] = tmpSysVars[j].toUpperCase();

//Uppercase system vars
for (int j=0; j<SystemVars.length; j++)
        SystemVars[j] = SystemVars[j].toUpperCase();

//Process variable
//String actualVar = templateVars[i][1];
String actualVar = "";
String actualVarTotal = "";
if (Parser.equals("FALSE")){
        //Process only one record
        String old = "{"+SysVars[0]+"}";
        String src = templateVars[i][1].toUpperCase();

        if (SysVars[0].startsWith(SystemVars[7]))
          if (Category != null){
                actualVarTotal = replace(src, old,
section.getLinkbyName(getParameter(SysVars[0]), Category));
                }else{
                actualVarTotal = replace(src, old,
section.getLinkbyName(getParameter(SysVars[0])));
                }
        else if (SysVars[0].startsWith(SystemVars[9])){
                actualVarTotal = replace(src, old,
article.getLinkbyName(getParameter(SysVars[0])));
                parse.append("dentro de getArticleLink... "+actualVarTotal +
"<br>\n");
        }else if (SysVars[0].equals(SystemVars[10]))
                if (getSiteName().equals("gedas"))
                        actualVarTotal = "index_gedas.jsp?ln="+LN;
                else
                        actualVarTotal = "index_vw.jsp?ln="+LN;
        else if (SysVars[0].equals(SystemVars[11]))
                if (getSiteName().equals("gedas"))
                        actualVarTotal = "sitemap_gedas.jsp?ln="+LN;
                else
                        actualVarTotal = "sitemap_vw.jsp?ln="+LN;
        else if (SysVars[0].equals(SystemVars[15]))
                if (getSiteName().equals("gedas"))
                        if (LN == 1)
                                actualVarTotal = "index_gedas.jsp?ln=2";
                        else
                                actualVarTotal = "index_gedas.jsp?ln=1";
                else
                        if (LN == 1)
                                actualVarTotal = "index_vw.jsp?ln=2";
                        else
                                actualVarTotal = "index_vw.jsp?ln=1";

                parse.append("Tpl var: "+src.length()+"
UserVar:"+old.length()+ " Category: "+Category+"<br>\n");
                parse.append("Var["+i+"] = "+src+"<br>\n");
```

```
                                        parse.append("Old: "+old+"<br>\n");
                                        parse.append("lynkBYName:
"+section.getLinkbyName(getParameter(SysVars[0]),Category)+"<br>\n");
                                        parse.append("Actual: "+actualVarTotal + "<br>\n");
                                        va.put(templateVars[i][0], actualVarTotal);
                                }else{
                                        //Process a list of records
                                        //get list of section's childs
                                        String[][] childs;
                                        parse.append("----------
----------<br>\n");
                                        parse.append(templateVars[i][0] + "<BR>\n");
                                        parse.append("----------
----------<br>\n");
                                        parse.append("Padre= "+Parent + " Category= " + Category + "
Language= "+ LN + " Counter= "+ Counter +"<br>\n");
                                        if (Category != null)
                                                if (getSiteName().equals("gedas"))
                                                        childs = section.getChilds(Parent, LN);
                                                else
                                                        childs = section.getChilds(Parent, LN, Category);
                                        else
                                                if (getSiteName().equals("gedas"))
                                                        childs = section.getChilds(Parent, LN);
                                                else
                                                        childs = section.getChilds(Parent, LN,
"MenuNavigation");


                                parse.append("antes de getArticles<br>\n");
                                        //Get list of articles
                                        String[] Arts = article.getArticles(LN, Parent);

                                        //get # records
                                        int records = 0;
                                        if (childs != null)
                                                records = childs.length;

                                        parse.append("Records: "+records+ " : " + (childs != null) +
"<br>\n");

                                        //check if system var require articles
                                        int sec_require = 0;
                                        for (int j=0; j<SysVars.length; j++)
                                                if (SysVars[j].indexOf("SECTION") != -1)
                                                sec_require = 1;

                                        parse.append("childs: "+records+"<br>\n");
                                        if (sec_require != 0){
                                                //Process each template var for all section records
                                                for (int j=0; j<records; j++){
                                                parse.append("child name: "+childs[j][1]+"<br>\n");
                                                actualVar = variabletoUpperCase(templateVars[i][1]);
                                                        for (int k=0; k<SysVars.length; k++){
                                                                if (SystemVars[1].equals(SysVars[k])){
                                                                        actualVar =
utilz.replaceAll(actualVar, "{"+SysVars[k]+"}", childs[j][1]);
                                                                        processed = true;
                                                                }else if
(SystemVars[2].equals(SysVars[k])){
```

```
                                                actualVar =
utilz.replaceAll(actualVar, "{"+SysVars[k]+"}", section.getLink(childs[j][0]));
                                                processed = true;
                                        }else if
(SystemVars[6].equals(SysVars[k])){
                                                actualVar =
utilz.replaceAll(actualVar, "{"+SysVars[k]+"}", ""+Counter);
                                                processed = true;
                                        }else if
(SysVars[k].startsWith(SystemVars[7])){
                                                actualVar =
utilz.replaceAll(actualVar, "{"+SysVars[k]+"}", section.getLinkbyName(getParameter(SysVars[k])));
                                                processed = true;
                                        }
                                }
                                Counter++;
                                actualVarTotal += actualVar;
                        }
                }

                //check if system var require articles
                int art_require = 0;
                for (int j=0; j<SysVars.length; j++)
                        if (SysVars[j].indexOf("ARTICLE") != -1)
                        art_require = 1;

                if ((art_require != 0) && (Arts != null)){
                        //Process each template var for all article records
                        for (int j=0; j<Arts.length; j++){
                          actualVar = variabletoUpperCase(templateVars[i][1]);
                                for (int k=0; k<SysVars.length; k++){
                                        parse.append("sysvar =
"+SysVars[k]+"<br>\n");
                                        if (SystemVars[3].equals(SysVars[k])){
                                                actualVar =
utilz.replaceAll(actualVar, "{"+SysVars[k]+"}", article.getTitle(Arts[j]));
                                                processed = true;
                                        }else if
(SystemVars[4].equals(SysVars[k])){
                                                actualVar =
utilz.replaceAll(actualVar, "{"+SysVars[k]+"}", article.getSummary(Arts[j]));
                                                processed = true;
                                        }else if
(SystemVars[5].equals(SysVars[k])){
                                                actualVar =
utilz.replaceAll(actualVar, "{"+SysVars[k]+"}", article.getLink(Arts[j]));
                                                processed = true;
                                        }else if
(SystemVars[6].equals(SysVars[k])){
                                                actualVar =
utilz.replaceAll(actualVar, "{"+SysVars[k]+"}", ""+Counter);
                                                processed = true;
                                        }
                                }
                                Counter++;
                                actualVarTotal += actualVar;
                        }
                }

                /*
                //Process system date
                for (int k=0; k<SysVars.length; k++){
```

```
                                        if (SystemVars[0].equals(SysVars[k])){
                                                if (actualVar.equals(""))
                                                        actualVar                                       =
templateVars[i][1].toUpperCase();
                                                if (actualVar.length() == 13)
                                                        actualVar = getDate();
                                                else
                                                        actualVar       =       replaceAll(actualVar,
"{"+SysVars[k]+"}", getDate());
                                                processed = true;
                                                break;
                                        }
                                }
                                */

                        }

                        //process general system Vars
                        String actualTpl = templateFile.substring(0, templateFile.indexOf("/"));
                        String  TemplatePath  =  utilz.replaceAll(AppPath,  "\\/",  "-")  +  "templates-"  +
actualTpl + "-";
                        TemplatePath = replaceAll(TemplatePath, "-", "\\/");
                        actualVarTotal      =      utilz.replaceAll(actualVarTotal,      "{TEMPLATEPATH}",
TemplatePath);
                        String ServletPath = utilz.replaceAll(Servlet, "\\/", "-");
                        ServletPath = replaceAll(ServletPath, "-", "\\/");
                        actualVarTotal      =      utilz.replaceAll(actualVarTotal,      "{SERVLETPATH}",
ServletPath);
                        String ApplicationPath = utilz.replaceAll(AppPath, "\\/", "-");
                        ApplicationPath = replaceAll(ApplicationPath, "-", "\\/");
                        actualVarTotal      =      utilz.replaceAll(actualVarTotal,      "{VIRTUALPATH}",
ApplicationPath);

                        VarInProcess += actualVarTotal;
                        if (processed)
                                va.put(templateVars[i][0], VarInProcess);

                        parse.append(templateVars[i][0] + " = " + VarInProcess + "<br>\n");
                }

                ////////////////////////
                //Process template's user variables
                ////////////////////////
                int count_txt=0, count_img=0, count_file=0;
                for (int i=0; i<templateUserVars.length; i++){
                  boolean processed = false;
                        String varInProcess = "";
                        parse.append(templateUserVars[i] + "<br>\n");
                        parse.append("texto: "+article_texto.size() + " imagen: " + article_imagen.size()
+ " file:"+article_file.size()+"<BR>\n");
                        if (templateUserVars[i].endsWith("_TXT")){
                                String texto = (String) article_texto.elementAt(count_txt++);
                                parse.append("texto: "+texto +"<BR>\n");
                                varInProcess = texto;
                                processed = true;
                        }else if (templateUserVars[i].endsWith("_IMG")){
                                String                  imagen                  =                  (String)
article_imagen.elementAt(count_img++);
                                varInProcess                            =                            "<IMG
SRC=\""+AppPath+"sections/"+imagen+"\" BORDER=O>\n";
                                parse.append("imagen: "+varInProcess +"<BR>\n");
                                processed = true;
```

```
                                        }else if (templateUserVars[i].endsWith("_FILE")){
                                                String file = (String) article_file.elementAt(count_file++);
                                                varInProcess = AppPath+"sections/"+file;
                                                parse.append("file: "+varInProcess +"<BR>\n");
                                                processed = true;
                                        }

                                        if (processed)
                                                va.put(templateUserVars[i], varInProcess);

                                }

                        }
                        catch(Exception e){
                          va.put("END_BODY", "Error, no se pudo mostrar la seccion. "+e.toString());
                                parse.append("Error: " + e.toString());
                                handleException(e, "getSection("+idSec+", "+idLanguage+") - Error - "+e.toString());
                        }finally{
                                try{
                                        connectionPool.free(connection);
                                        handleException(ex, "getSection("+idSec+", "+idLanguage+") - closed -
"+connectionPool.toString());
                                }catch(SQLException sql){
                                        writeLog("No se pudeo cerrar conexion en: getSection("+idSec+",
"+idLanguage+")");
                                }

                                int pos = templateFile.indexOf("/");
                                String Tmp;
                                if (pos != -1)
                                Tmp = templateFile.substring(0, pos);
                                else
                                  Tmp = templateFile;
                                String Tpl = AppPath + "templates/" + Tmp;

                         template.set_vars(va);
                         //String parse = template.print_template(templateFile, Tmp);
                                parse.delete(0, parse.capacity());
                                parse.append(template.print_template(templateFile, Tmp));
                                long end_time = System.currentTimeMillis();


                                parse.append("Tiempo total: "+ (end_time-init_time) );

                                if (parse == null || parse.capacity() == 0){
                                        parse.delete(0, parse.capacity());
                                parse.append(template.last_error);
                                }

                                setCacheData(cache, parse.toString());
                                return( parse.toString() );
                        }
                        }
 }
```

Figura 45. Código para agregar desplegar secciones en Front End

Así como hay un método para las secciones también se cuenta con un método para los artículos, los cuales, al ejecutar el método también estarán listos para desplegarse como una página HTML. Para más detalle veamos el siguinete código:

```
public String getArticle(String idArt, String idLanguage){
  GnaSvtTemplate template = new GnaSvtTemplate(TplPath);
        Map va = new HashMap();
        String templateFile="", query, parse = "";
        String tpl, idSec;
        boolean dispacherexists = false;
        Vector section_id = new Vector();
        Vector section_parent = new Vector();
        Vector section_language = new Vector();
        Vector section_name = new Vector();
        Vector section_idCat = new Vector();
        Vector section_nameCat = new Vector();
        Vector article_id = new Vector();
        Vector article_section = new Vector();
        Vector article_title = new Vector();
        Vector article_summary = new Vector();
        Vector article_dispacher = new Vector();
        Vector article_language = new Vector();
        Vector article_texto = new Vector();
        Vector article_imagen = new Vector();
        Vector article_file = new Vector();

        String cache = "getArticle:" + idArt + ";" + idLanguage;
        if ((user_request != null) && (cacheContains(cache))){
          writeLog(connectionPool.toString());
            return ( (String)getCacheData(cache) );
        }else{

        Connection connection = null;
        Exception ex = new Exception("Error");
  try {
    connection = connectionPool.getConnection();
            handleException(ex, "getArticle("+idArt+", "+idLanguage+") -
"+connectionPool.toString());
        //get id template
        //query = "select id_template, id_seccion from pg_articulo where id_articulo="+idArt;
        //query = "select id_template, id_seccion from pg_articulo with(nolock) where id_articulo="+idArt;
        //DBResults results = DatabaseUtilities.getQueryResults(connection,query, false);

        query = "select id_template, id_seccion from pg_articulo with(nolock) where id_articulo=?";
            String[] params = {idArt};
            DBResults results = DatabaseUtilities.getQueryResults(connection, query, params, false);

    //String[] data = new String[2];
            String[] data;
      data = results.getRow(0);
            tpl = data[0];
            idSec = data[1];

        //get template file
        //query = "select archivo from pg_template where id_template="+tpl;
        //query = "select archivo from pg_template with(nolock) where id_template="+tpl;
```

```
                //results = DatabaseUtilities.getQueryResults(connection,query, false);

                query = "select archivo from pg_template with(nolock) where id_template=?";
                        params[0] = tpl;
                        results = DatabaseUtilities.getQueryResults(connection, query, params, false);

        data = new String[1];
          data = results.getRow(0);
                        templateFile = data[0];

                //get variable data
                //query = "select nombre, contenido from pg_variable where nombre not like ('%_TXT') and nombre not like ('%_IMG') and nombre not like ('%_FILE') and id_template="+tpl;
                //query = "select nombre, contenido from pg_variable with(nolock) where nombre not like ('%_TXT') and nombre not like ('%_IMG') and nombre not like ('%_FILE') and id_template="+tpl;
                //results = DatabaseUtilities.getQueryResults(connection,query, false);

                query = "select nombre, contenido from pg_variable with(nolock) where nombre not like ('%_TXT') and nombre not like ('%_IMG') and nombre not like ('%_FILE') and id_template=?";
                        params[0] = tpl;
                        results = DatabaseUtilities.getQueryResults(connection, query, params, false);

                        int rows = results.getRowCount();

                        String[][] templateVars = new String[rows][2];
                        for (int i=0; i<rows; i++){
          data = new String[2];
            data = results.getRow(i);
                                templateVars[i][0] = data[0];
                                templateVars[i][1] = data[1];
                        }

                //get user variable data
                //query = "select nombre from pg_variable where (nombre like ('%_TXT') or nombre like ('%_IMG') or nombre like ('%_FILE')) and id_template="+tpl;
                //query = "select nombre from pg_variable with(nolock) where (nombre like ('%_TXT') or nombre like ('%_IMG') or nombre like ('%_FILE')) and id_template="+tpl;
                //results = DatabaseUtilities.getQueryResults(connection,query, false);

                query = "select nombre from pg_variable with(nolock) where (nombre like ('%_TXT') or nombre like ('%_IMG') or nombre like ('%_FILE')) and id_template=?";
                        params[0] = tpl;
                        results = DatabaseUtilities.getQueryResults(connection, query, params, false);

                        rows = results.getRowCount();

                        String[] templateUserVars = new String[rows];
                        for (int i=0; i<rows; i++){
          data = new String[1];
            data = results.getRow(i);
                                templateUserVars[i] = data[0];
                        }

                //get articles data
                //query = "select id_articulo, id_seccion, id_idioma, titulo, resumen, dispacher from pg_articulo where id_seccion is not null and id_idioma="+idLanguage+" order by id_articulo, id_seccion";
                //query = "select id_articulo, id_seccion, id_idioma, titulo, resumen, dispacher from pg_articulo with(nolock) where id_seccion is not null and id_idioma="+idLanguage+" order by id_articulo, id_seccion";
                //results = DatabaseUtilities.getQueryResults(connection,query, false);

                query = "select id_articulo, id_seccion, id_idioma, titulo, resumen, dispacher from pg_articulo with(nolock) where id_seccion is not null and id_idioma=? order by id_articulo, id_seccion";
```

```
                        params[0] = idLanguage;
                        results = DatabaseUtilities.getQueryResults(connection, query, params, false);

                        rows = results.getRowCount();

                        for (int i=0; i<rows; i++){
            data = new String[6];
              data = results.getRow(i);
                                article_id.add(data[0]);
                                if (data[1] == null)
                                        article_section.add("NULL");
                                else
                                        article_section.add(data[1]);
                                article_language.add(data[2]);
                                article_title.add(data[3]);
                                article_summary.add(data[4]);
                                article_dispacher.add(data[5]);
                        }

                //get sections data
                //query = "select s.id_seccion, s.secc_padre, s.id_idioma, s.descripcion, c.id_cat, cat.descripcion from pg_seccion s, pg_secc_cat c, pg_cat cat where s.id_seccion = c.id_seccion and c.id_cat = cat.id_cat and s.id_idioma="+idLanguage+" and s.id_seccion <> 0 order by s.id_seccion";
                //query = "select s.id_seccion, s.secc_padre, s.id_idioma, s.descripcion, c.id_cat, cat.descripcion from pg_seccion s, pg_secc_cat c, pg_cat cat with(nolock) where s.id_seccion = c.id_seccion and c.id_cat = cat.id_cat and s.id_idioma="+idLanguage+" and s.id_seccion <> 0 order by s.id_seccion";
                //results = DatabaseUtilities.getQueryResults(connection,query, false);

                query = "select s.id_seccion, s.secc_padre, s.id_idioma, s.descripcion, c.id_cat, cat.descripcion from pg_seccion s, pg_secc_cat c, pg_cat cat with(nolock) where s.id_seccion = c.id_seccion and c.id_cat = cat.id_cat and s.id_idioma=? and s.id_seccion <> 0 order by s.id_seccion";
                        params[0] = idLanguage;
                        results = DatabaseUtilities.getQueryResults(connection, query, params, false);

                        rows = results.getRowCount();

                        for (int i=0; i<rows; i++){
            data = new String[6];
              data = results.getRow(i);
                                section_id.add(data[0]);
                                if (data[1] == null)
                                        section_parent.add("NULL");
                                else
                                        section_parent.add(data[1]);
                                section_language.add(data[2]);
                                section_name.add(data[3]);
                                section_idCat.add(data[4]);
                                section_nameCat.add(data[5]);
                        }

                        //Main section object
                        GnaObjSection section = new GnaObjSection(section_id, section_parent, section_language, section_name, section_idCat, section_nameCat);
                        //Main article object
                        GnaObjArticle article = new GnaObjArticle(article_id, article_section, article_language, article_title, article_summary, article_dispacher);

                        ////////////////////////////////////
                        //Get Section texts, images and files
                        //get section's texto
                //query = "select texto from pg_texto where id_articulo="+idArt+" order by id_texto";
                //query = "select texto from pg_texto with(nolock) where id_articulo="+idArt+" order by id_texto";
```

```
                //results = DatabaseUtilities.getQueryResults(connection, query, false);

                query = "select texto from pg_texto with(nolock) where id_articulo=? order by id_texto";
                        params[0] = idArt;
                        results = DatabaseUtilities.getQueryResults(connection, query, params, false);

                        rows = results.getRowCount();

                        for (int i=0; i<rows; i++){
        data = new String[1];
          data = results.getRow(i);
                        article_texto.add(data[0]);
                        }

                        //get section's texto
                //query = "select path from pg_imagen where id_articulo="+idArt+" order by id_imagen";
                //query = "select path from pg_imagen with(nolock) where id_articulo="+idArt+" order by
id_imagen";
                //results = DatabaseUtilities.getQueryResults(connection, query, false);

                query = "select path from pg_imagen with(nolock) where id_articulo=? order by id_imagen";
                        params[0] = idArt;
                        results = DatabaseUtilities.getQueryResults(connection, query, params, false);

                        rows = results.getRowCount();

                        for (int i=0; i<rows; i++){
        data = new String[1];
          data = results.getRow(i);
                        article_imagen.add(data[0]);
                        }

                        //get section's texto
                //query = "select path from pg_file where id_articulo="+idArt+" order by id_file";
                //query = "select path from pg_file with(nolock) where id_articulo="+idArt+" order by id_file";
                //results = DatabaseUtilities.getQueryResults(connection, query, false);

                query = "select path from pg_file with(nolock) where id_articulo=? order by id_file";
                        params[0] = idArt;
                        results = DatabaseUtilities.getQueryResults(connection, query, params, true);  //connection
closed

                        rows = results.getRowCount();

                        for (int i=0; i<rows; i++){
        data = new String[1];
          data = results.getRow(i);
                        article_file.add(data[0]);
                        }

                        //get Directives & SystemVars from template's variables
                        Vector[] templateDirectives = new Vector[templateVars.length];
                        Vector[] templateVariables  = new Vector[templateVars.length];
                        for (int i=0; i<templateVars.length; i++){
                                String[] directivesArray = new String[countDirectives(templateVars[i][1])];
                                directivesArray = getDirectives(templateVars[i][1]);

                                Map directives = new HashMap();
                                for (int j=0; j<directivesArray.length; j++)
                                  directives.put(directivesArray[j],              getDirectiveValue(templateVars[i][1],
directivesArray[j]));
```

```
                    String[] systemVars = new String[countSysVars(templateVars[i][1])];
                    systemVars = getSystemVars(templateVars[i][1]);
                    templateDirectives[i] = new Vector();
                    templateDirectives[i].add(directives);  //Directives list of each template's variable
                    templateVariables[i] = new Vector();
                    templateVariables[i].add(systemVars);    //System variables list of each template's variable
                    templateVars[i][1]  =  eraseDirectives(templateVars[i][1]);    //delete  directives from template's variable
                }

                parse += templateVars.length + " total<bR>";
                ////////////////////////
                //Process template's system variables
                ////////////////////////
                for (int i=0; i<templateVars.length; i++){
                    //Init directives
                    int Counter = 0;
                    String Languaje = "ESPAÑOL";
                    String Category = null;
                    String Parent   = null;
                    String Parser   = "TRUE";
                    int LN = GnaObjSection.SPANISH;

                    String VarInProcess = "";
                    boolean processed = false;

                    //Process directives
                    HashMap dirs = (HashMap) templateDirectives[i].elementAt(0);
                    if (dirs.containsKey("LANGUAGE")) Languaje = (String) dirs.get("LANGUAGE");
                    if (dirs.containsKey("CATEGORY")) Category = (String) dirs.get("CATEGORY");
                    if (dirs.containsKey("PARENT"))   Parent = (String) dirs.get("PARENT");
                    if (dirs.containsKey("COUNTER")) Counter = Integer.parseInt(dirs.get("COUNTER").toString());
                    if (dirs.containsKey("PARSER"))   Parser = (String) dirs.get("PARSER");

                    if (Languaje.equals("ESPAÑOL"))
                        LN = GnaObjSection.SPANISH;
                    else if (Languaje.equals("INGLES"))
                        LN = GnaObjSection.ENGLISH;
                    //LN = idLanguage;

                    if (Parent == null)
                        Parent = idSec;
                    else if (Parent.equalsIgnoreCase("Home"))
                      Parent = "NULL";
                    else if (Parent.equalsIgnoreCase("{"+ SystemVars[12] +"}")){
                        String padre = section.getParent(idSec);
                        Parent = padre;
                    }else if (Parent.equalsIgnoreCase("{"+ SystemVars[13] +"}")){
                        String padre = article.getParent(idArt);
                        Parent = padre;
                    }else
                        Parent = section.getID(Parent, idLanguage);

                    //Get System variables from template's variable
                    String[] tmpSysVars = (String[]) templateVariables[i].elementAt(0);
                    int count=0,pos=0;
                    for (int j=0; j<tmpSysVars.length; j++)
                        if (tmpSysVars[j] != null)
```

```
                    count++;

                String[] SysVars = new String[count];
                for (int j=0; j<tmpSysVars.length; j++)
                        if (tmpSysVars[j] != null)
                                SysVars[pos++] = tmpSysVars[j].toUpperCase();

                //Uppercase system vars
                for (int j=0; j<SystemVars.length; j++)
                        SystemVars[j] = SystemVars[j].toUpperCase();

                //Process variable
                //String actualVar = templateVars[i][1];
                String actualVar = "";
                String actualVarTotal = "";
                if (Parser.equals("FALSE")){
                        //Process only one record
                        String old = "{"+SysVars[0]+"}";
                        String src = templateVars[i][1].toUpperCase();

                        if (SysVars[0].startsWith(SystemVars[7]))
                          if (Category != null)
                                        actualVarTotal = replace(src, old,
section.getLinkbyName(getParameter(SysVars[0]), Category));
                                else
                                        actualVarTotal = replace(src, old,
section.getLinkbyName(getParameter(SysVars[0])));
                        else if (SysVars[0].equals(SystemVars[10]))
                                if (getSiteName().equals("gedas"))
                                        actualVarTotal = "index_gedas.jsp?ln="+LN;
                                else
                                        actualVarTotal = "index_vw.jsp?ln="+LN;
                        else if (SysVars[0].equals(SystemVars[11]))
                                if (getSiteName().equals("gedas"))
                                        actualVarTotal = "sitemap_gedas.jsp?ln="+LN;
                                else
                                        actualVarTotal = "sitemap_vw.jsp?ln="+LN;
                        else if (SysVars[0].equals(SystemVars[15]))
                                if (getSiteName().equals("gedas"))
                                        if (LN == 1)
                                                actualVarTotal = "index_gedas.jsp?ln=2";
                                        else
                                                actualVarTotal = "index_gedas.jsp?ln=1";
                                else
                                        if (LN == 1)
                                                actualVarTotal = "index_vw.jsp?ln=2";
                                        else
                                                actualVarTotal = "index_vw.jsp?ln=1";


                                parse += "Tpl var: "+src.length()+"
UserVar:"+old.length()+ " Category: "+Category+"<br>\n";
                                parse += "Var["+i+"] = "+src+"<br>\n";
                                parse += "Old: "+old+"<br>\n";
                                parse += "lynkBYName:
"+section.getLinkbyName(getParameter(SysVars[0]), Category)+"<br>\n";
                                parse += "Actual: "+actualVarTotal + "<br>\n";
                                va.put(templateVars[i][0], actualVarTotal);
                }else{
                        //Process a list of records
                        //get list of section's childs
                        String[][] childs;
```

```
                                        parse += "----------------------------------------------------------------------------
-----<br>\n";
                                        parse += templateVars[i][0] + "<BR>\n";
                                        parse += "----------------------------------------------------------------------------
-----<br>\n";
                                        parse += "Padre= "+Parent + " Category= " + Category + "
Language= "+ LN + " Counter= "+ Counter +"<br>\n";
                                        if (Category != null)
                                                if (getSiteName().equals("gedas"))
                                                        childs = section.getChilds(Parent, LN);
                                                else
                                                        childs = section.getChilds(Parent, LN, Category);
                                        else
                                                if (getSiteName().equals("gedas"))
                                                        childs = section.getChilds(Parent, LN);
                                                else
                                                        childs      =      section.getChilds(Parent,      LN,
"MenuNavigation");

                                parse += "antes de getArticles<br>\n";
                                        //Get list of articles
                                        String[] Arts = article.getArticles(LN, Parent);

                                        //get # records
                                        int records = 0;
                                        if (childs != null)
                                                records = childs.length;

                                        parse += "Records: "+records+ " : " + (childs != null) + "<br>\n";



                                        //check if system var require section
                                        int sec_require = 0;
                                        for (int j=0; j<SysVars.length; j++)
                                                if (SysVars[j].indexOf("SECTION") != -1)
                                                sec_require = 1;

                                        parse += "childs: "+records+"<br>\n";
                                        if (sec_require != 0){
                                                //Process each template var for all section records
                                                for (int j=0; j<records; j++){
                                                parse += "child name: "+childs[j][1]+"<br>\n";
                                                actualVar = variabletoUpperCase(templateVars[i][1]);
                                                        for (int k=0; k<SysVars.length; k++){
                                                                if (SystemVars[1].equals(SysVars[k])){
                                                                        actualVar                       =
utilz.replaceAll(actualVar, "{"+SysVars[k]+"}", childs[j][1]);
                                                                        processed = true;
                                                                }else                                   if
(SystemVars[2].equals(SysVars[k])){
                                                                        actualVar                       =
utilz.replaceAll(actualVar, "{"+SysVars[k]+"}", section.getLink(childs[j][0]));
                                                                        processed = true;
                                                                }else                                   if
(SystemVars[6].equals(SysVars[k])){
                                                                        actualVar                       =
utilz.replaceAll(actualVar, "{"+SysVars[k]+"}", ""+Counter);
                                                                        processed = true;
                                                                }else                                   if
(SysVars[k].startsWith(SystemVars[7])){
```

```
                                                actualVar =
utilz.replaceAll(actualVar, "{"+SysVars[k]+"}", section.getLinkbyName(getParameter(SysVars[k])));
                                                processed = true;
                                            }
                                        }
                                        Counter++;
                                        actualVarTotal += actualVar;
                                    }
                                }

                                //check if system var require articles
                                int art_require = 0;
                                for (int j=0; j<SysVars.length; j++)
                                    if (SysVars[j].indexOf("ARTICLE") != -1)
                                    art_require = 1;

                                //if ((art_require != 0) && (Arts != null)){
                                if (Arts != null){
                                    //Process each template var for all article records
                                    for (int j=0; j<Arts.length; j++){
                                      actualVar = variabletoUpperCase(templateVars[i][1]);
                                        for (int k=0; k<SysVars.length; k++){
                                            parse += "sysvar =
"+SysVars[k]+"<br>\n";
                                            if (SystemVars[3].equals(SysVars[k])){
                                                actualVar =
utilz.replaceAll(actualVar, "{"+SysVars[k]+"}", article.getTitle(Arts[j]));
                                                processed = true;
                                            }else if
(SystemVars[4].equals(SysVars[k])){
                                                actualVar =
utilz.replaceAll(actualVar, "{"+SysVars[k]+"}", article.getSummary(Arts[j]));
                                                processed = true;
                                            }else if
(SystemVars[5].equals(SysVars[k])){
                                                actualVar =
utilz.replaceAll(actualVar, "{"+SysVars[k]+"}", article.getLink(Arts[j]));
                                                processed = true;
                                            }else if
(SystemVars[6].equals(SysVars[k])){
                                                actualVar =
utilz.replaceAll(actualVar, "{"+SysVars[k]+"}", ""+Counter);
                                                processed = true;
                                            }else if
(SystemVars[14].equals(SysVars[k])){
                                             String dis = article.getDispacher(idArt);
                                                if (!dis.equals("")){
                                                  page = dis;
                                                   dispacherexists = true;
                                                }
                                            }
                                        }
                                        Counter++;
                                        actualVarTotal += actualVar;
                                    }
                                }

                                /*
                                //Process system date
                                for (int k=0; k<SysVars.length; k++){
                                    if (SystemVars[0].equals(SysVars[k])){
                                        if (actualVar.equals(""))
```

```
                                        actualVar =
templateVars[i][1].toUpperCase();
                                        if (actualVar.length() == 13)
                                                actualVar = getDate();
                                        else
                                                actualVar = replaceAll(actualVar,
"{"+SysVars[k]+"}", getDate());
                                        processed = true;
                                        break;
                                }
                        }
                        */

                }

                //process general system Vars
                String actualTpl = templateFile.substring(0, templateFile.indexOf("/"));
                String TemplatePath = utilz.replaceAll(AppPath, "\\", "-") + "templates-" +
actualTpl + "-";
                TemplatePath = replaceAll(TemplatePath, "-", "\\");
                actualVarTotal = utilz.replaceAll(actualVarTotal, "{TEMPLATEPATH}",
TemplatePath);
                String ServletPath = utilz.replaceAll(Servlet, "\\", "-");
                ServletPath = replaceAll(ServletPath, "-", "\\");
                actualVarTotal = utilz.replaceAll(actualVarTotal, "{SERVLETPATH}",
ServletPath);
                String ApplicationPath = utilz.replaceAll(AppPath, "\\", "-");
                ApplicationPath = replaceAll(ApplicationPath, "-", "\\");
                actualVarTotal = utilz.replaceAll(actualVarTotal, "{VIRTUALPATH}",
ApplicationPath);

                VarInProcess += actualVarTotal;
                if (processed)
                        va.put(templateVars[i][0], VarInProcess);

                parse += templateVars[i][0] + " = " + VarInProcess + "<br>\n";
        }

        //////////////////////
        //Process template's user variables
        //////////////////////
        int count_txt=0, count_img=0, count_file=0;
        for (int i=0; i<templateUserVars.length; i++){
          boolean processed = false;
                String varInProcess = "";
                parse += templateUserVars[i] + "<br>\n";
                parse += "texto: "+article_texto.size() + " imagen: " + article_imagen.size() + "
file:"+article_file.size()+"<BR>\n";
                if (templateUserVars[i].endsWith("_TXT")){
                        String texto = (String) article_texto.elementAt(count_txt++);
                        parse += "texto: "+texto +"<BR>\n";
                        varInProcess = texto;
                        processed = true;
                }else if (templateUserVars[i].endsWith("_IMG")){
                        String imagen = (String)
article_imagen.elementAt(count_img++);
                        varInProcess = "<IMG
SRC=\""+AppPath+"articles/"+imagen+"\" BORDER=O>\n";
                        parse += "imagen: "+varInProcess +"<BR>\n";
                        processed = true;
                }else if (templateUserVars[i].endsWith("_FILE")){
                        String file = (String) article_file.elementAt(count_file++);
```

```
                                                varInProcess = "<A HREF=\""+AppPath+"articles/"+file+"\">Download</A>\n";
                                                parse += "file: "+varInProcess +"<BR>\n";
                                                processed = true;
                                        }

                                        if (processed)
                                                va.put(templateUserVars[i], varInProcess);

                                }

                        }
                        catch(Exception e){
                          va.put("END_BODY", "Error, no se pudo mostrar el articulo. "+e.toString());
                                parse += "Error: " + e.toString();
                                handleException(e, "getArticle("+idArt+", "+idLanguage+") - Error - "+e.toString());
                        }finally{
                                try{
                                        connectionPool.free(connection);
                                        handleException(ex, "getArticle("+idArt+", "+idLanguage+") - closed - "+connectionPool.toString());
                                }catch(SQLException sql){
                                        writeLog("No se pudeo cerrar conexion en: getArticle("+idArt+", "+idLanguage+")");
                                }

                                int pos = templateFile.indexOf("/");
                                String Tmp;
                                if (pos != -1)
                                Tmp = templateFile.substring(0, pos);
                                else
                                  Tmp = templateFile;
                                String Tpl = AppPath + "templates/" + Tmp;

                         template.set_vars(va);
                         //String parse = template.print_template(templateFile, Tmp);
                                parse = template.print_template(templateFile, Tmp);

                                if (dispacherexists){
                                        dispacher = true;
                                }
                                if (parse == null)
                                parse = template.last_error;

                                if (!useDispacher())
                                        setCacheData(cache, parse);
                                return parse;
                        }
                        }
 }
```

Figura 46. Código para desplegar artículos en Front End

Las clases de GnaBeansObjArticle y GnaBeansObjSections únicamente se encargan de tomar datos correspondientes a los artículos y a las secciones, me refiero a datos como lo

son el nombre de la sección o artículo, la sección a la que pertenece un artículo o una subsección y demás. A continuación se mostrará el diagrama de clase para cada uno de ellos.

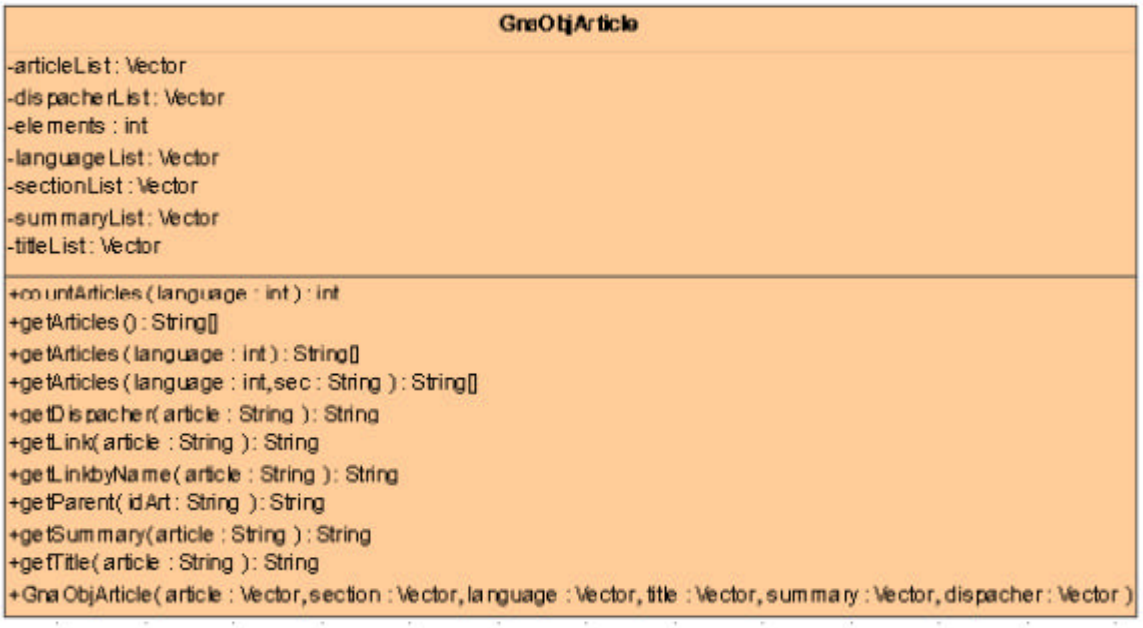

Figura 47. Diagrama de clase para GnaObjArticle.

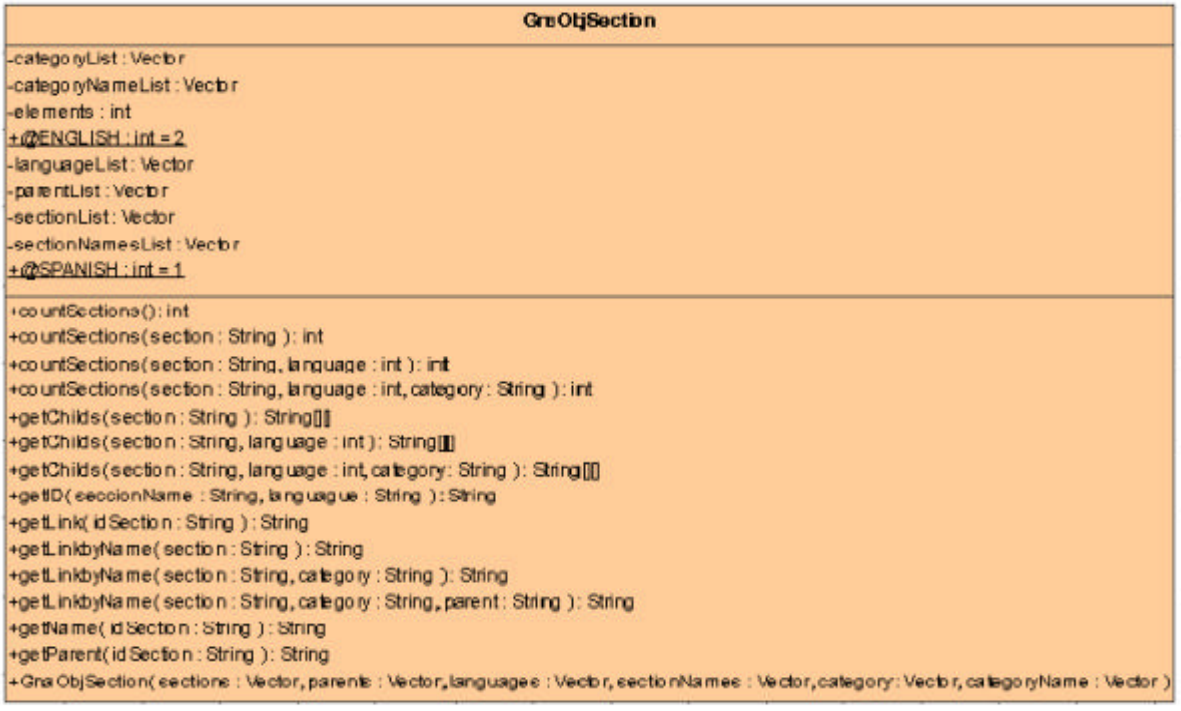

Figura 48. Diagrama de clase para GnaObjSection

Podemos observar que cada uno de estos Beans crean las ligas entre secciones y artículos, es decir, se encargan de formar el árbol de navegación del sitio.

IV.2.2 Controller.

Por parte del controller del Front End tenemos al archivos llamado index.jsp", el cual no sólo actúa como controller, sino que también actúa como parte del view ya que es la página principal del sistema. Es el controller ya que es el encargado de mandar a llamar los Java Beans anteriormente mencionados y los despliega como página web, esto gracias a que los Java Server Pages (JSP) son una combinación de código HTML y código Java.

Como se menciono en el párrafo anterior el JSP llamado index forma una parte importante del view, ya que es la página principal del sitio, pero la mayor parte del View del Front End esta formado por cada una de las secciones y artículos construidos en el Back End, ya que estos son los que aquellos que accedan al sitio ven y navegan.

La librería de Perl [Perl for Java] será utilizada en el Front End ya que permitirá que aquellas páginas del sitio que contengan variables del sistema sean sustituidas por el valor correspondiente; un ejemplo de ello es una variable que se llama Date, esta variable va a tener el valor de la fecha en la que se despliegue el sitio.

IV.3. Estructura de Carpetas dentro del Servidor

Para el funcionamiento del proyecto es necesario tanto de un personal Web Server como puede ser el Personal Web Server de Microsoft, el Apache Web Server o alguno otro.

También se necesita un "engine" para ejecutar Servlets como puede ser el ServletExec ISAPI 3.1 [Servlet Exec ISAPI] o el Tomcat de Apache [Apache Tomcat]. Todo esto además de las librerías Perl y O´Reilly anteriormente mencionadas. La aplicación debe estar dentro de la carpeta Webapps y debe contener la estructura que se muestra en la siguiente figura:

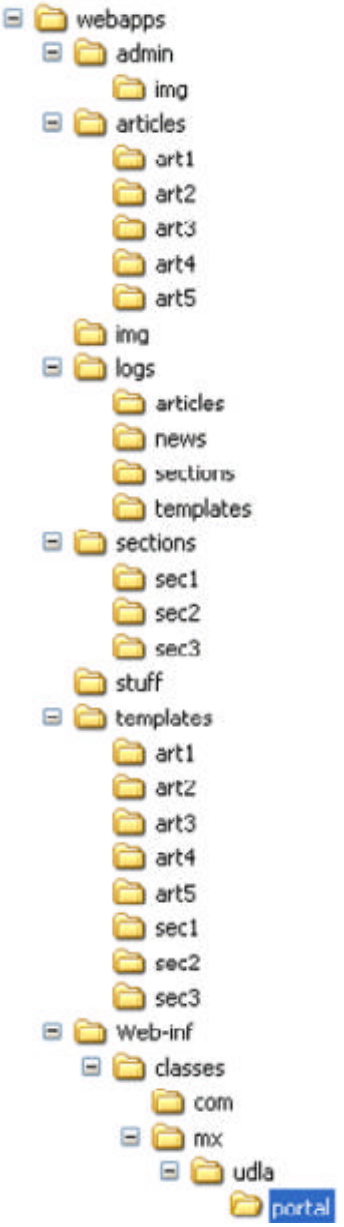

Figura 49. Estructura de directorios dentro del webapps en Servlet-Exec

El fólder admin contendrá los templates de todas las páginas utilizadas en el BackEnd. Además contendrá un fólder llamado img en el cual estarán todas las imágenes utilizadas por las páginas del BackEnd. Tal como se puede apreciar en la siguiente figura:

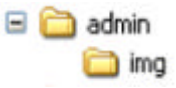

Figura 50. Estructura de directorios para templates.

La carpeta articles contendrá las páginas de cada uno de los artículos creados. Cada artículo tendrá su propia carpeta, por ejemplo el articulo1 tendrá una carpeta llamada art1, dentro de la cual estarán los archivos del artículo, como lo son el archivo HTML, archivos javascripts utilizados por el artículo y además una carpeta llamada img que contendrá las imágenes del artículo. La estructura de la carpeta articles se muestra gráficamente en la siguiente figura:

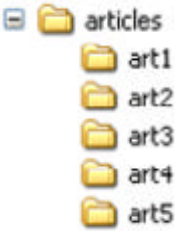

Figura 51. Estructura de directorios para articles.

Existe una carpeta llamada img, la cual contiene las imágenes utilizadas por los jsp del index del sitio.

Figura 52. Estructura de directorios para las imágenes de la página principal del sitio.

La siguiente carpeta dentro de la estructura es la carpeta de logs, la cual contiene archivos de texto con los errores que han ocurrido dentro del sistema en el momento de la construcción de un sitio o cuando se consulta el sitio construido.

Figura 53. Estructura de directorios para los Logs.

La carpeta de sections tiene una estructura muy similar a la carpeta de articles, ya que dentro de esta existen otras carpetas con el nombre de la sección, por ejemplo: sec1, sec2, sec3, etcétera. Estas subcarpetas contienen dentro los archivos utilizados por la sección, como lo son el archivo HTML, archivos javascripts utilizados por la sección y además un fólder llamada img en el cual están todas las imágenes utilizadas por la sección.

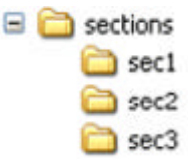

Figura 54. Estructura de directorios para las Secciones.

La carpeta templates contiene subcarpetas que contienen los diferentes templates tanto para una sección como para un artículo. Estas subcarpetas contendrán nombres como art1, art1, …, artN o sec1, sec2, …, secN. Dentro de cada una de las subcarpetas están tanto los archivos utilizados por el template como los son el HTML, javascripts, hojas de estilos (css) y una subcarpeta llamada img, dentro de la cual están todas las imágenes utilizadas por el template.

Figura 55. Estructura de directorios para Templates.

Finalmente la carpeta llamada Web-inf contiene un package con todos los archivos Java y las clases Java utilizadas por el sistema, además contiene el driver que permite la comunicación con la base de datos.

Figura 56. Estructura de directorios para las clases java del sistema.

Dentro de este proyecto se construyeron las herramientas de secciones, artículos, templates y noticias, las cuales cada una de ellas muestra la información que existe dentro de la base de datos y además muestra las opciones de agregar, eliminar, modificar e incluso en algunos casos ver la información final, es decir, como se verá al navegar un sitio.

Figura 57. Pantalla principal del BackEnd.

Cada una de las herramientas requiere de cierta información para poder agregar o modificar a la base de datos, para ello se valida que la información corresponda con el tipo de datos de la base de datos (mediante javascript), en otras palabras, sí en la base de datos el nombre del nuevo artículo, sección, template o noticia es un tipo de dato varchar y el usuario introduce solo números, el sistema mandará un mensaje de error. De igual manera se valida que se introduzca la información mínima necesaria para agregar o modificar información. En el caso de eliminar información se valida que esta información no sea

compartida con algún otro módulo o que no este en uso porque en este caso se lanzará un mensaje de error indicando quien esta utilizando la información, también es necesario que se confirme que la información quiere ser eliminada, ya que una vez eliminada de la base de datos la información no podrá ser recuperada

En el caso de modificar información también se valida que la información no sea utilizada por alguna otra herramienta ya que se perderían las relaciones y causaría problemas dentro del sitio.

Otro punto muy importante dentro de la aplicación es la manera de atender las transacciones ya que cada transacción requiere accesos a la base de datos y es necesario que la información sea desplegada de manera rápida para que el usuario no se desespere, ni pierda interés en el sitio. Para atender las peticiones y las conexiones con la base de datos se desarrollo un Pool de Conexiones, el cual por default siempre mantiene 10 conexiones abiertas y un total de 50 conexiones para cuando las 10 conexiones iniciales estén ocupadas, el propósito es que una vez que un usuario desocupe una, otro usuario pueda tomarla y así agilizar el despliegue de datos.

Todo lo que se refiere al BackEnd fue desarrollado con componentes Java Servlets y existe un Servlet del cual heredan todas las demás clases, este Servlet se llama PortalServlet y es el encargado de establecer la conexión con la base datos, esta instancia a otros componentes como lo es el Pool de Conexiones y otras clases importantes. La razón por la cual se desarrollo la clase PortalServlet es porque evitamos que cada vez que haya una petición se abra una nueva conexión con la base de datos, con la clase PortalServlet se

obliga a que cuando exista una nueva petición se revise cuantas conexiones de las 10 conexiones abiertas por default están abiertas, si aún existen conexiones abiertas se utiliza una de ellas, si no, se abrirá una nueva conexión la cual será cerrada cuando sea necesario y de esta manera se mantendrá un equilibrio de conexiones entre el sistema y la base de datos. Cabe mencionar que el Pool de conexiones esta diseñado para soportar alrededor de 200 usuarios concurrentes, en dado caso que el sitio requiera un mayor número de usuarios concurrentes únicamente se deberá aumentar el número de conexiones totales a utilizar. La siguiente figura muestra de manera gráfica la manera de trabajar del Pool de Conexiones.

El propósito de la explicación de cada uno de los componentes que forman el modelo "Model View Controller" tanto para el Back End como para el Front End es que conozca como fueron desarrolladas cada una de las clases y cual es la relación que existen entre ellas y con la base de datos, para que en el momento que se este software se quiera extender o modificar, se conozca en que parte se debe modificar el código.

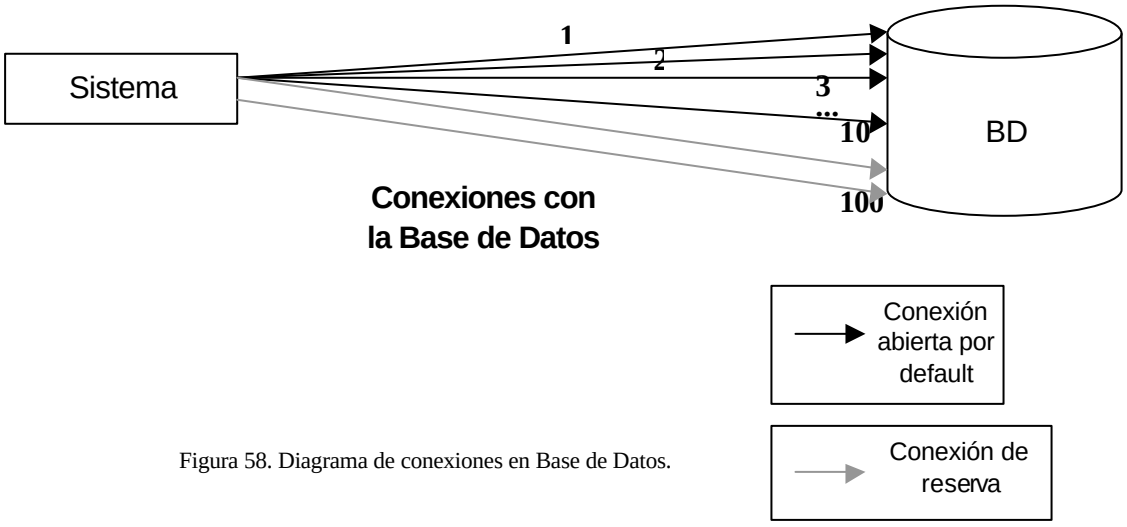

Figura 58. Diagrama de conexiones en Base de Datos.