Javier Isaías Coyoy Marín 202030556

Manual Técnico

Servidor

Para el servidor utilizamos socket, el servidor es el encargado de recibir los textos que son escritos tanto en el área de XML como en el área de Consultas, y por medio de otro mensaje que recibe el servidor lo casteamos a int podemos obtener cuál de las dos acciones debemos realizar, para el análisis del contenido del texto utilizamos los analizadores léxico y sintáctico que se detallaran más adelante.

Un dato importante es que el servidor es inicializado en otro hilo, que se inicia al momento de iniciar toda la aplicación, con el objetivo que siempre este presente durante toda la vida útil del programa, luego al realizar el analizar respectivo envía mensajes al cliente para que pueda saber si su operación fue exitosa, por ultimo manda instrucciones al manejar de páginas para que cree, modifique o elimine según sean las acciones escritas, así como realizar consultas en base a lo solicitado.

```
public class Servidor extends Thread {

    private final int PUERTO = 80;

    @Override
    public void run() {
        ServerSocket servidor = null;
        Socket socket = null;
        DataInputStream entrada;
        DataOutputStream salida;

        try {
            servidor = new ServerSocket(PUERTO);

            while(true) {
                socket = servidor.accept();
                entrada = new DataInputStream(socket.getInputStream());
                salida = new DataOutputStream(socket.getOutputStream());

                String contenidoDelPanel = entrada.readUTF();
                int operacion = Integer.parseInt(entrada.readUTF());
                String mensaje = null;

                if (operacion == 1) {
```

```
                    LexicoXML lexico = new LexicoXML(new
StringReader(contenidoDelPanel));
                    SintacticoXML parser = new SintacticoXML(lexico);

                    try {
                        parser.parse();
                        mensaje = "Todo esta Correcto, realizando Acciones";
                        GeneradorHtml generador = new GeneradorHtml();
                        generador.analizarTokens(contenidoDelPanel);
                        ArchivoBaseDeDatos generador2 = new ArchivoBaseDeDatos();
                        generador2.crearArchivoBaseDeDatos(contenidoDelPanel);
                    } catch (Exception ex) {
                        Symbol sym = parser.getSymbol();

                        if (sym != null) {
                            switch (sym.sym) {
                                case 0 -> mensaje = "Error Sintáctico en la linea
" + sym.left + ", columna " + sym.right + ": Se esperaba el Resto de la
Instrucción";
                                default -> mensaje = "Error Sintáctico en la
linea " + sym.left + ", columna " + sym.right + ": " + parser.getContenido();
                            }
                        }
                    } catch (Error ex){
                        mensaje = ex.getMessage();
                    }
                } else if (operacion == 2){
                    LexicoConsultas lexico = new LexicoConsultas(new
StringReader(contenidoDelPanel));
                    SintacticoConsultas parser = new SintacticoConsultas(lexico);

                    try {
                        parser.parse();
                        mensaje = "Todo esta Correcto, realizando Consultas";
                    } catch (Exception ex) {
                        Symbol sym = parser.getSymbol();

                        if (sym != null) {
                            switch (sym.sym) {
                                case 0 -> mensaje = "Error Sintáctico en la linea
" + sym.left + ", columna " + sym.right + ": Se esperaba el Resto de la
Instrucción";
                                default -> mensaje = "Error Sintáctico en la
linea " + sym.left + ", columna " + sym.right + ": " + parser.getContenido();
                            }
```

```
                        }
                    } catch (Error ex){
                        mensaje = ex.getMessage();
                    }
                }

                salida.writeUTF(mensaje);
                socket.close();
            }
        } catch(IOException ex) {
            System.out.println("Sucedio una Excepción de tipo: " +
ex.getMessage());
        }
    }
}
```

ClienteXML

Esta clase es utilizada para mandar información al servidor, en este caso mandamos todo lo que está contenido en el panel de XML, y lo manda para que pueda ser analizado y si todo está correcto se ejecute las acciones establecidas, mientras tanto el cliente se queda a la espera de la respuesta del servidor que en este caso es un mensaje el cual se mostrara en el panel de resultados de la aplicación con el objetivo que el usuario este enterado, y si en caso hubiera errores se pueda guiar para que lo pueda corregir.

```
public class ClienteXML {

    private final String HOST = "127.0.0.1";
    private final int PUERTO = 80;

    public void conectar(String contenidoDelPanel, JTextPane areaDeConsola) {
        DataInputStream entrada;
        DataOutputStream salida;

        try {
            Socket socket = new Socket(HOST,PUERTO);
            entrada = new DataInputStream(socket.getInputStream());
            salida = new DataOutputStream(socket.getOutputStream());

            salida.writeUTF(contenidoDelPanel);
            salida.writeUTF("1");

            String mensaje = entrada.readUTF();
```

```
            areaDeConsola.setText(mensaje);
            socket.close();
        } catch(IOException ex) {
            System.out.println("Sucedio una Excepción de tipo: " +
ex.getMessage());
        }
    }
}
```

ClienteConsultas

Esta clase es utilizada para lo mismo que en el apartado anterior con la única diferencia que lo que envía al servidor es el contenido del panel de consultas y de igual manera espera al mensaje del servidor para que le pueda mostrar al usuario el resultado de sus acciones.

Además, que adicional del mensaje del panel mandamos un segundo mensaje con el número de la acción que deseamos realizar, en este caso el 2 es para las consultas.

```
public class ClienteConsultas {

    private final String HOST = "127.0.0.1";
    private final int PUERTO = 80;

    public void conectar(String contenidoDelPanel, JTextPane areaDeConsola) {
        DataInputStream entrada;
        DataOutputStream salida;
        try {
            Socket socket = new Socket(HOST,PUERTO);
            entrada = new DataInputStream(socket.getInputStream());
            salida = new DataOutputStream(socket.getOutputStream());

            salida.writeUTF(contenidoDelPanel);
            salida.writeUTF("2");

            String mensaje = entrada.readUTF();
            areaDeConsola.setText(mensaje);
            socket.close();
        } catch(IOException ex) {
            System.out.println("Sucedio una Excepción de tipo: " +
ex.getMessage());
        }
    }
}
```

Fronted

Dentro del fronted no tenemos mucha lógica, ya que la única función es estar ahí para recibir las instrucciones del usuario y mandar esa formación por medio de las clases de cliente, además de métodos privados que nos sirven para obtener la información de los paneles.

Y otro dato importante es que desde aquí es donde se inicializa el hilo del servidor para que esté presente durante toda la ejecución del programa.

```
private void botonAnalizarYEjecutarActionPerformed(java.awt.event.ActionEvent
evt) {
        if (!"".equals(obtenerTexto())) {
            ClienteXML cliente = new ClienteXML();
            cliente.conectar(obtenerTexto(), areaDeConsola);
        } else {
            JOptionPane.showMessageDialog(this,"Debe Ingresar Información para
Analizar","Error",JOptionPane.ERROR_MESSAGE);
        }
    }

    private void botonConsultaActionPerformed(java.awt.event.ActionEvent evt)
{
        if (!"".equals(obtenerConsulta())) {
            ClienteConsultas cliente = new ClienteConsultas();
            cliente.conectar(obtenerConsulta(), areaDeConsola);
        } else {
            JOptionPane.showMessageDialog(this,"Debe Ingresar Información para
Consultar","Error",JOptionPane.ERROR_MESSAGE);
        }
    }

    private void iniciarServidor() {
        servidor.start();
    }

    private String obtenerTexto() {
        return areaDeTexto.getText();
    }

    private String obtenerConsulta() {
        return areaDeConsultas.getText();
    }
```

Populares

Esta clase fue creada para poder guardar la información de la página y del número de visitas que tiene con el objetivo de poder obtener solo el número de visitas al momento de realizar el ordenamiento de los datos

```
package com.mycompany.aplicacioncliente.main.consultas;

public class Populares {

    private String pagina;
    private int vistas;

    public Populares(String pagina, int vistas) {
        this.pagina = pagina;
        this.vistas = vistas;
    }

    public String getPagina() {
        return pagina;
    }

    public int getVistas() {
        return vistas;
    }
}
```

FrameConsultas

Esta clase instancia un jFrame que nos va servir para mostrar los resultados de las consultas que el usuario haga, como parámetro inicial recibimos la lista de elementos que conforman la respuesta y creamos un nuevo modelo de tabla que nos va servir para mostrarle los resultados al usuario.

```
    private void mostrarConsultas() {
        tablaConsultas.removeAll();
        String[] titulos = {resultados.get(0)};
        tablaModelo.setColumnIdentifiers(titulos);
        tablaConsultas.setModel(tablaModelo);

        for (int i = 1; i < resultados.size(); i++) {
            tablaModelo.addRow(new Object[] {resultados.get(i), i, 1});
        }
    }
```

Manejador De Paginas

Esta clase se encarga de crear, modificar y eliminar los sitios y paginas html, por medio de una colección personalizada de tokens el programa va poder determinar qué acción debe guardar y realizarla dentro del documento preseleccionado con el objetivo de ejemplificar el manejo de una base de datos con la cual vamos a poder visualizar y manipular datos.

Analizar Tokens

Este primer método es utilizado para el reconocimiento de acciones. Por medio del analizador léxico con el cual obtenemos todos los tokens y con ayuda del siguiente código vamos a poder determinar cada una de las acciones que deseamos realizar y luego de encontrar los elementos que necesitamos procedemos a guardarlos dentro de una lista y se lo mandamos al siguiente método.

```
public void analizarTokens(String contenido) {
        LexicoXML lexico = new LexicoXML(new StringReader(contenido));
        List<String> listaTokens = new ArrayList<>();

        try {
            while (!lexico.yyatEOF()) {
                Symbol symbol = lexico.next_token();

                if (sym.NUEVO_SITIO == symbol.sym) {
                    if (sym.NUEVO_SITIO == symbol.sym && !listaTokens.isEmpty())
{
                        crearArchivoBaseDeDatos(listaTokens);
                        listaTokens = new ArrayList<>();
                        marcarTodosFalse();
                    }
                    nuevoSitioActivo = true;
                } else if (sym.BORRAR_SITIO == symbol.sym) {
                    if (sym.BORRAR_SITIO == symbol.sym && !listaTokens.isEmpty())
{
                        crearArchivoBaseDeDatos(listaTokens);
                        listaTokens = new ArrayList<>();
                        marcarTodosFalse();
                    }
                    eliminarSitioActivo = true;
                } else if (sym.NUEVA_PAGINA == symbol.sym) {
                    if (sym.NUEVA_PAGINA == symbol.sym && !listaTokens.isEmpty())
{
                        crearArchivoBaseDeDatos(listaTokens);
                        listaTokens = new ArrayList<>();
```

```
                    marcarTodosFalse();
                }
                nuevaPaginaActiva = true;
            } else if (sym.MODIFICAR_PAGINA == symbol.sym) {
                if (sym.MODIFICAR_PAGINA == symbol.sym &&
!listaTokens.isEmpty()) {
                    crearArchivoBaseDeDatos(listaTokens);
                    listaTokens = new ArrayList<>();
                    marcarTodosFalse();
                }
                modificarPaginaActiva = true;
            } else if (sym.BORRAR_PAGINA == symbol.sym) {
                if (sym.BORRAR_PAGINA == symbol.sym &&
!listaTokens.isEmpty()) {
                    crearArchivoBaseDeDatos(listaTokens);
                    listaTokens = new ArrayList<>();
                    marcarTodosFalse();
                }
                eliminarPaginaActiva = true;
            } else if (sym.AGREGAR_COMPONENTE == symbol.sym ||
sym.MODIFICAR_COMPONENTE == symbol.sym) {
                if ((sym.AGREGAR_COMPONENTE == symbol.sym ||
sym.MODIFICAR_COMPONENTE == symbol.sym) && !listaTokens.isEmpty()) {
                    crearArchivoBaseDeDatos(listaTokens);
                    listaTokens = new ArrayList<>();
                    marcarTodosFalse();
                }
                amComponenteActivo = true;
            } else if (sym.BORRAR_COMPONENTE == symbol.sym) {
                if ((sym.BORRAR_COMPONENTE == symbol.sym) &&
!listaTokens.isEmpty()) {
                    crearArchivoBaseDeDatos(listaTokens);
                    listaTokens = new ArrayList<>();
                    marcarTodosFalse();
                }
                eliminarComponenteActivo = true;
            }

            if (nuevoSitioActivo || nuevaPaginaActiva) {
                if (sym.PALABRA == symbol.sym || sym.FECHA == symbol.sym ||
sym.FECHA_C == symbol.sym || sym.FECHA_M == symbol.sym || sym.USUARIO_M ==
symbol.sym) {
                    listaTokens.add(symbol.value.toString());
                }
```

```
                } else if (modificarPaginaActiva || eliminarSitioActivo ||
eliminarPaginaActiva || amComponenteActivo || eliminarComponenteActivo) {
                    if (sym.PALABRA == symbol.sym) {
                        listaTokens.add(symbol.value.toString());
                    }
                }
            }
            if (!listaTokens.isEmpty()) {
                crearArchivoBaseDeDatos(listaTokens);
            }
        } catch (IOException ex) {
            // Manejo de Error
        }
    }
```

Crear Archivo Base De Datos

Luego este segundo método es utilizado para realizar CRUD al archivo de texto que tenemos establecido como base de datos en los casos particulares de los componentes en el apartado de modificación se crea utilizando el usuario anónimo. Por otro lado, cuando se trata de páginas nuevas se agrega al usuario especificado en la acción.

Otro dato importante es que el modo de separar cada sitio es por medio de líneas en blanco, por su parte para agregar páginas se busca la línea punteada del sitio y se agrega al final de la misma con información abstraída de la acción.

```
private void crearArchivoBaseDeDatos(List<String> listaTokens) {
        List<String> lineas = new ArrayList<>();
        boolean escribir = false;

        if (nuevoSitioActivo) {
            lineas.add("Sitio=" + listaTokens.get(0));
            String creacion = listaTokens.get(1) + ",";
            int contador = 2;

            if (contador < listaTokens.size()) {
                if ("FECHA_CREACION".equals(listaTokens.get(contador))) {
                    contador++;
                    creacion += listaTokens.get(contador);
                    contador++;
                } else {
                    creacion += obtenerFecha();
                }
```

```
            } else {
                creacion += obtenerFecha();
            }
            lineas.add("Creacion=" + creacion);
            String fechaModificacion = "";

            if (contador < listaTokens.size()) {
                if ("FECHA_MODIFICACION".equals(listaTokens.get(contador))) {
                    contador++;
                    fechaModificacion += listaTokens.get(contador);
                    contador++;
                } else {
                    fechaModificacion += obtenerFecha();
                }
            } else {
                fechaModificacion += obtenerFecha();
            }
            String usuarioModificacion = "";

            if (contador < listaTokens.size()) {
                if ("USUARIO_MODIFICACION".equals(listaTokens.get(contador))) {
                    contador++;
                    usuarioModificacion += listaTokens.get(contador);
                    contador++;
                } else {
                    usuarioModificacion += listaTokens.get(1);
                }
            } else {
                usuarioModificacion += listaTokens.get(1);
            }
            lineas.add("Modificacion=" + usuarioModificacion + "," +
fechaModificacion);
            lineas.add("------------------------------");
            lineas.add("");
            guardarArchivoBaseDeDatos(lineas, true);
        } else if (eliminarSitioActivo) {
            List<String> todasLasLineas = obtenerArchivoBaseDeDatos();
            int indice = 0;

            while (indice < todasLasLineas.size()) {
                String separar[] = todasLasLineas.get(indice).split("=");

                if ("Sitio".equals(separar[0]) &&
separar[1].equals(listaTokens.get(0))) {
```

```
                    while (!"".equals(todasLasLineas.get(indice))) {
                        todasLasLineas.remove(indice);
                    }
                }
                indice++;
            }
            guardarArchivoBaseDeDatos(todasLasLineas, false);
        } else if (nuevaPaginaActiva) {
            List<String> todasLasLineas = obtenerArchivoBaseDeDatos();
            int indice = 0, contador = 5;

            lineas.add("Pagina=" + listaTokens.get(0));
            String creacion = listaTokens.get(4) + ",";

            if (contador < listaTokens.size()) {
                if ("FECHA_CREACION".equals(listaTokens.get(contador))) {
                    contador++;
                    creacion += listaTokens.get(contador);
                    contador++;
                } else {
                    creacion += obtenerFecha();
                }
            } else {
                creacion += obtenerFecha();
            }
            lineas.add("Creacion=" + creacion);
            String fechaModificacion = "";

            if (contador < listaTokens.size()) {
                if ("USUARIO_MODIFICACION".equals(listaTokens.get(contador))) {
                    contador++;
                    usuarioModificacion += listaTokens.get(contador);
                    contador++;
                } else {
                    usuarioModificacion += listaTokens.get(4);
                }
            } else {
                usuarioModificacion += listaTokens.get(4);
            }
            String modificacion = "Modificacion=" + usuarioModificacion + "," +
fechaModificacion;
            lineas.add(modificacion);
            lineas.add("Vistas=0");
            lineas.add("------------------------------");
```

```
            while (indice < todasLasLineas.size()) {
                String separar[] = todasLasLineas.get(indice).split("=");

                if ("Sitio".equals(separar[0]) &&
separar[1].equals(listaTokens.get(2))) {
                    agregarModificacionAlSitio(todasLasLineas,
listaTokens.get(2), modificacion);

                    while (!"------------------------------
".equals(todasLasLineas.get(indice))) {
                        indice++;
                    }
                    indice++;

                    int limite = (lineas.size() - 1);
                    for (int i = limite; i >= 0; i--) {
                        todasLasLineas.add(indice, lineas.get(i));
                    }
                    break;
                }
                indice++;
            }
            guardarArchivoBaseDeDatos(todasLasLineas, false);
        } else if (modificarPaginaActiva) {
            List<String> todasLasLineas = obtenerArchivoBaseDeDatos();
            String sitioNombre = "", modificacion = "Modificacion=Anonimo," +
obtenerFecha();
            int indice = 0;

            while (indice < todasLasLineas.size()) {
                String separar[] = todasLasLineas.get(indice).split("=");

                if ("Sitio".equals(separar[0])) {
                    sitioNombre = separar[1];
                    escribir = true;
                }
                if ("Pagina".equals(separar[0]) &&
separar[1].equals(listaTokens.get(0))) {
                    indice += 2;
                    todasLasLineas.remove(indice);
                    todasLasLineas.add(indice, modificacion);
                    break;
                }
                indice++;
            }
```

```
            if (escribir) {
                agregarModificacionAlSitio(todasLasLineas, sitioNombre,
modificacion);
            }
            guardarArchivoBaseDeDatos(todasLasLineas, false);
        } else if (eliminarPaginaActiva) {
            List<String> todasLasLineas = obtenerArchivoBaseDeDatos();
            String sitioNombre = "", modificacion = "Modificacion=Anonimo," +
obtenerFecha();
            int indice = 0;

            while (indice < todasLasLineas.size()) {
                String separar[] = todasLasLineas.get(indice).split("=");

                if ("Sitio".equals(separar[0])) {
                    sitioNombre = separar[1];
                    escribir = true;
                }
                if ("Pagina".equals(separar[0]) &&
separar[1].equals(listaTokens.get(0))) {

                    while (!"--------------------------------
".equals(todasLasLineas.get(indice))) {
                        todasLasLineas.remove(indice);
                    }
                    todasLasLineas.remove(indice);
                    break;
                }
                indice++;
            }
            if (escribir) {
                agregarModificacionAlSitio(todasLasLineas, sitioNombre,
modificacion);
            }
            guardarArchivoBaseDeDatos(todasLasLineas, false);
        } else if (amComponenteActivo) {
            List<String> todasLasLineas = obtenerArchivoBaseDeDatos();
            String sitioNombre = "", modificacion = "Modificacion=Anonimo," +
obtenerFecha();
            int indice = 0;

            while (indice < todasLasLineas.size()) {
                String separar[] = todasLasLineas.get(indice).split("=");

                if ("Sitio".equals(separar[0])) {
```

```
                    sitioNombre = separar[1];
                    escribir = true;
                }
                if ("Pagina".equals(separar[0]) &&
separar[1].equals(listaTokens.get(1))) {
                    break;
                }
                indice++;
            }
            if (escribir) {
                agregarModificacionAlSitio(todasLasLineas, sitioNombre,
modificacion);
            }
            guardarArchivoBaseDeDatos(todasLasLineas, false);
        } else if (eliminarComponenteActivo) {
            List<String> todasLasLineas = obtenerArchivoBaseDeDatos();
            String sitioNombre = "", modificacion = "Modificacion=Anonimo," +
obtenerFecha();
            int indice = 0;

            while (indice < todasLasLineas.size()) {
                String separar[] = todasLasLineas.get(indice).split("=");

                if ("Sitio".equals(separar[0])) {
                    sitioNombre = separar[1];
                    escribir = true;
                }
                if ("Pagina".equals(separar[0]) &&
separar[1].equals(listaTokens.get(0))) {
                    break;
                }
                indice++;
            }
            if (escribir) {
                agregarModificacionAlSitio(todasLasLineas, sitioNombre,
modificacion);
            }
            guardarArchivoBaseDeDatos(todasLasLineas, false);
        }
    }
```

Gramáticas

GramaticaXML

Esta gramática es utilizada para validar los archivos xml que el usuario ingrese en el apartado correspondiente, como primer paso definimos el analizador léxico de la siguiente manera:

Primero definimos el lugar donde colocaríamos la clase, luego definimos características importantes como el cup para conectarse con el analizador sintáctico, la línea y la columna para obtener los números en caso de notificar errores, después definimos las expresiones regulares para algunos de los casos que se nos presentan al momento de analizar el archivo y luego definimos en cada uno de los casos que se nos presenten con que identificador vamos a mandar el token al analizador sintáctico y ese seria en general el trabajo que se hizo dentro de este analizador.

```
package com.mycompany.aplicacioncliente.main.analizadoresxml;

import java_cup.runtime.*;

%% // Separador de Area

%class LexicoXML
%public
%cup
%line
%column
%eofval{
    return symbol(sym.EOF);
%eofval}

%{
    private Symbol symbol(int type) {
        return new Symbol(type, yyline+1, yycolumn+1);
    }

    private Symbol symbol(int type, Object value) {
        return new Symbol(type, yyline+1, yycolumn+1, value);
    }
%}

Saltos = \r|\n|\r\n
Espacios = {Saltos} | [ \t\f]
Atributos = [aA][tT][rR][iI][bB][uU][tT][oO][sS]
```

```
Etiquetas = [eE][tT][iI][qQ][uU][eE][tT][aA][sS]
Palabra = [[_]|[-]|[$]][[a-zA-Z]|[0-9]][[a-zA-Z][0-9]]*
Contenido = [a-zA-Z][a-zA-Z]*
Color = [#][[a-zA-Z]|[0-9]]*
Url = [h][t][t][p][s][[a-zA-Z]|[0-9]|[-]|[_]|[ ]|[.]|[/]|[?]|[=]|[:]|[@]]*
Fecha = [0-9][0-9][0-9][0-9][/][0-9][0-9][/][0-9][0-9]
Numero = [0-9][0-9]*

%% // Separador de Area

<YYINITIAL> {

    NUEVO_SITIO_WEB       { return symbol(sym.NUEVO_SITIO, yytext()); }

    NUEVA_PAGINA          { return symbol(sym.NUEVA_PAGINA, yytext()); }

    BORRAR_SITIO_WEB      { return symbol(sym.BORRAR_SITIO, yytext()); }

    BORRAR_PAGINA         { return symbol(sym.BORRAR_PAGINA, yytext()); }

    MODIFICAR_PAGINA      { return symbol(sym.MODIFICAR_PAGINA, yytext()); }

    AGREGAR_COMPONENTE    { return symbol(sym.AGREGAR_COMPONENTE, yytext()); }

    BORRAR_COMPONENTE     { return symbol(sym.BORRAR_COMPONENTE, yytext()); }

    MODIFICAR_COMPONENTE  { return symbol(sym.MODIFICAR_COMPONENTE, yytext()); }

    ID                    { return symbol(sym.ID, yytext()); }

    TITULO                { return symbol(sym.TITULO, yytext()); }

    SITIO                 { return symbol(sym.SITIO, yytext()); }

    PADRE                 { return symbol(sym.PADRE, yytext()); }

    PARRAFO               { return symbol(sym.PARRAFO, yytext()); }

    IMAGEN                { return symbol(sym.IMAGEN, yytext()); }

    VIDEO                 { return symbol(sym.VIDEO, yytext()); }

    MENU                  { return symbol(sym.MENU, yytext()); }

    USUARIO_CREACION      { return symbol(sym.USUARIO_C, yytext()); }
```

```
FECHA_CREACION        { return symbol(sym.FECHA_C, yytext()); }
FECHA_MODIFICACION    { return symbol(sym.FECHA_M, yytext()); }
USUARIO_MODIFICACION  { return symbol(sym.USUARIO_M, yytext()); }
PAGINA                { return symbol(sym.PAGINA, yytext()); }
CLASE                 { return symbol(sym.CLASE, yytext()); }
TEXTO                 { return symbol(sym.TEXTO, yytext()); }
ALINEACION            { return symbol(sym.ALINEACION, yytext()); }
IZQUIERDA             { return symbol(sym.IZQUIERDA, yytext()); }
CENTRAR               { return symbol(sym.CENTRAR, yytext()); }
DERECHA               { return symbol(sym.DERECHA, yytext()); }
COLOR                 { return symbol(sym.COLOR, yytext()); }
ALTURA                { return symbol(sym.ALTURA, yytext()); }
ANCHO                 { return symbol(sym.ANCHO, yytext()); }
COLOR                 { return symbol(sym.COLOR, yytext()); }
ORIGEN                { return symbol(sym.ORIGEN, yytext()); }
acciones              { return symbol(sym.ACCIONES, yytext()); }
accion                { return symbol(sym.ACCION, yytext()); }
parametros            { return symbol(sym.PARAMETROS, yytext()); }
parametro             { return symbol(sym.PARAMETRO, yytext()); }
atributos             { return symbol(sym.ATRIBUTOS, yytext()); }
atributo              { return symbol(sym.ATRIBUTO, yytext()); }
etiqueta              { return symbol(sym.ETIQUETA, yytext()); }
```

```
    nombre                  { return symbol(sym.NOMBRE, yytext()); }

    valor                  { return symbol(sym.VALOR, yytext()); }

    "\""                    { return symbol(sym.COMILLA, yytext()); }

    "<"                     { return symbol(sym.MENOR, yytext()); }

    "</"                { return symbol(sym.MENOR_DIAGONAL, yytext()); }

    ">"                     { return symbol(sym.MAYOR, yytext()); }

    "/>"                { return symbol(sym.MAYOR_DIAGONAL, yytext()); }

    "["                     { return symbol(sym.CORCHETE_I, yytext()); }

    "]"                     { return symbol(sym.CORCHETE_D, yytext()); }

    "="                     { return symbol(sym.IGUAL, yytext()); }

    {Atributos}             { return symbol(sym.ATRIBUTOS, yytext()); }

    {Etiquetas}             { return symbol(sym.ETIQUETAS, yytext()); }

    {Palabra}               { return symbol(sym.PALABRA, yytext()); }

    {Contenido}             { return symbol(sym.CONTENIDO, yytext()); }

    {Color}             { return symbol(sym.NUMERO_COLOR, yytext()); }

    {Url}               { return symbol(sym.URL, yytext()); }

    {Fecha}             { return symbol(sym.FECHA, yytext()); }

    {Numero}                { return symbol(sym.NUMERO, yytext()); }

    {Espacios}              {/* ignoramos */}
}

[^]                         { throw new Error("Error Léxico caracter Invalido en la
linea " + (yyline+1) + ", columna " + (yycolumn+1) + ": " + yytext()); }
```

Como segunda parte de la gramatica tenemos el analizador sintáctico que para este proyecto fue necesario el uso de cup para poder realizarlo, como primer paso dentro de cup definimos algunos métodos que nos va permitir el retorno la información del token y el control de los posibles errores que sucedan dentro de la ejecución del programa.

Luego de esto empieza la parte de la gramatica donde primero definimos nuestros terminales que son todos aquellos tokens que definimos en la sección anterior, luego definimos los no terminales que son producciones que van a producir terminales y por ultimo viene lo más difícil que es definir la gramatica completa que nos garantice que cualquier acción que nos manden este correcta. Y si en caso hubiera un error se enviar un mensaje.

```
package com.mycompany.aplicacioncliente.main.analizadoresxml;

import java_cup.runtime.*;

parser code {:

    private Symbol symbol;
    private String contenido;

    public SintacticoXML(LexicoXML lexico) {
        super(lexico);
    }

    public void syntax_error(Symbol cur_token) {
        symbol = cur_token;
        contenido = (String)(cur_token.value);
    }

    public Symbol getSymbol() {
        return this.symbol;
    }

    public String getContenido() {
        return this.contenido;
    }
:}


/* Terminales */
terminal            NUEVO_SITIO, NUEVA_PAGINA, BORRAR_SITIO, BORRAR_PAGINA,
MODIFICAR_PAGINA, AGREGAR_COMPONENTE, BORRAR_COMPONENTE, MODIFICAR_COMPONENTE,
                    ID, TITULO, SITIO, PADRE, USUARIO_C, FECHA_C, FECHA_M,
USUARIO_M, PAGINA, CLASE,
```

```
                    ACCIONES, ACCION, PARAMETROS, PARAMETRO, ATRIBUTOS, ATRIBUTO,
ETIQUETAS, ETIQUETA,
                    PARRAFO, IMAGEN, VIDEO, MENU, TEXTO, ALINEACION, IZQUIERDA,
CENTRAR, DERECHA, COLOR, ORIGEN, ALTURA, ANCHO,
                    NOMBRE, VALOR,
                    COMILLA, MENOR, MENOR_DIAGONAL, MAYOR, MAYOR_DIAGONAL,
CORCHETE_I, CORCHETE_D, IGUAL;
terminal String     PALABRA, URL, FECHA, NUMERO_COLOR, CONTENIDO;
terminal int        NUMERO;

/* No Terminales */
non terminal        inicio, actividad, operacion,
                    inicioNuevoSitio, id1, usuario1, fecha1, fechaModificacion1,
usuarioModificacion1,
                    inicioNuevaPagina, id2, titulo, sitio, padre, usuario2,
fecha2, fechaModificacion2, usuarioModificacion2,
                    inicioModificarPagina, id3, tituloModificar, etiquetas,
etiqueta,
                    inicioBorrarSitio,
                    inicioBorrarPagina, id4,
                    inicioBorrarComponente, id5, pagina1,
                    inicioAgregarOModificarComponente, id6, pagina2, clase, tipo,
atributos1, atributos2, atributos3, atributo1, atributo2, atributo3, atributo4,
atributo5, atributo6, atributo7, atributo8,
                    contexto, contenido, direccion, fecha, numero, color;




/* Gramatica */
start with inicio;

inicio ::= MENOR ACCIONES MAYOR actividad MENOR_DIAGONAL ACCIONES MAYOR
;

actividad ::= MENOR ACCION NOMBRE IGUAL COMILLA operacion MENOR_DIAGONAL ACCION
MAYOR
        | MENOR ACCION NOMBRE IGUAL COMILLA operacion MENOR_DIAGONAL ACCION MAYOR
actividad
        ;

operacion ::= NUEVO_SITIO COMILLA MAYOR inicioNuevoSitio
        | NUEVA_PAGINA COMILLA MAYOR inicioNuevaPagina
        | BORRAR_SITIO COMILLA MAYOR inicioBorrarSitio
```

```
        | BORRAR_PAGINA COMILLA MAYOR inicioBorrarPagina
        | MODIFICAR_PAGINA COMILLA MAYOR inicioModificarPagina
        | AGREGAR_COMPONENTE COMILLA MAYOR inicioAgregarOModificarComponente
        | BORRAR_COMPONENTE COMILLA MAYOR inicioBorrarComponente
        | MODIFICAR_COMPONENTE COMILLA MAYOR inicioAgregarOModificarComponente
        ;









/* ------------------ Nuevo Sitio Web ------------------ */
inicioNuevoSitio ::= MENOR PARAMETROS MAYOR id1 MENOR_DIAGONAL PARAMETROS MAYOR
;

id1 ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA ID COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR usuario1
;

usuario1 ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA USUARIO_C COMILLA MAYOR
contexto MENOR_DIAGONAL PARAMETRO MAYOR fecha1
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA USUARIO_C COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR fechaModificacion1
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA USUARIO_C COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR usuarioModificacion1
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA USUARIO_C COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR
        ;

fecha1 ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA FECHA_C COMILLA MAYOR fecha
MENOR_DIAGONAL PARAMETRO MAYOR fechaModificacion1
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA FECHA_C COMILLA MAYOR fecha
MENOR_DIAGONAL PARAMETRO MAYOR usuarioModificacion1
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA FECHA_C COMILLA MAYOR fecha
MENOR_DIAGONAL PARAMETRO MAYOR
        ;

fechaModificacion1 ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA FECHA_M COMILLA MAYOR
fecha MENOR_DIAGONAL PARAMETRO MAYOR usuarioModificacion1
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA FECHA_M COMILLA MAYOR fecha
MENOR_DIAGONAL PARAMETRO MAYOR
        ;
```

```
usuarioModificacion1 ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA USUARIO_M COMILLA
MAYOR contexto MENOR_DIAGONAL PARAMETRO MAYOR
;




/* ------------------ Nueva Pagina Web ------------------ */
inicioNuevaPagina ::= MENOR PARAMETROS MAYOR id2
;

id2 ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA ID COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR titulo
;

titulo ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA TITULO COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR sitio
;

sitio ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA SITIO COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR padre
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA SITIO COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR usuario2
        ;

padre ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA PADRE COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR usuario2
;

usuario2 ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA USUARIO_C COMILLA MAYOR
contexto MENOR_DIAGONAL PARAMETRO MAYOR fecha2
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA USUARIO_C COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR fechaModificacion2
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA USUARIO_C COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR usuarioModificacion2
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA USUARIO_C COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR etiquetas
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA USUARIO_C COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR MENOR_DIAGONAL PARAMETROS MAYOR
        ;
```

```
fecha2 ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA FECHA_C COMILLA MAYOR fecha
MENOR_DIAGONAL PARAMETRO MAYOR fechaModificacion2
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA FECHA_C COMILLA MAYOR fecha
MENOR_DIAGONAL PARAMETRO MAYOR usuarioModificacion2
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA FECHA_C COMILLA MAYOR fecha
MENOR_DIAGONAL PARAMETRO MAYOR etiquetas
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA FECHA_C COMILLA MAYOR fecha
MENOR_DIAGONAL PARAMETRO MAYOR MENOR_DIAGONAL PARAMETROS MAYOR
        ;

fechaModificacion2 ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA FECHA_M COMILLA MAYOR
fecha MENOR_DIAGONAL PARAMETRO MAYOR usuarioModificacion2
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA FECHA_M COMILLA MAYOR fecha
MENOR_DIAGONAL PARAMETRO MAYOR etiquetas
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA FECHA_M COMILLA MAYOR fecha
MENOR_DIAGONAL PARAMETRO MAYOR MENOR_DIAGONAL PARAMETROS MAYOR
        ;

usuarioModificacion2 ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA USUARIO_M COMILLA
MAYOR contexto MENOR_DIAGONAL PARAMETRO MAYOR MENOR_DIAGONAL PARAMETROS MAYOR
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA USUARIO_M COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR etiquetas
        ;









/* ------------------ Modificar Pagina Web ------------------ */
inicioModificarPagina ::= MENOR PARAMETROS MAYOR id3
;

id3 ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA ID COMILLA MAYOR CORCHETE_I PALABRA
CORCHETE_D MENOR_DIAGONAL PARAMETRO MAYOR tituloModificar
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA ID COMILLA MAYOR CORCHETE_I
PALABRA CORCHETE_D MENOR_DIAGONAL PARAMETRO MAYOR etiquetas
        ;

tituloModificar ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA TITULO COMILLA MAYOR
CORCHETE_I PALABRA CORCHETE_D MENOR_DIAGONAL PARAMETRO MAYOR MENOR_DIAGONAL
PARAMETROS MAYOR
        | MENOR PARAMETRO NOMBRE IGUAL COMILLA TITULO COMILLA MAYOR CORCHETE_I
PALABRA CORCHETE_D MENOR_DIAGONAL PARAMETRO MAYOR etiquetas
```

```
        ;

/* ------------------ Parte en común entre Crear Pagina y Modificar Pagina ------
------------ */
etiquetas ::= MENOR_DIAGONAL PARAMETROS MAYOR MENOR ETIQUETAS MAYOR etiqueta
MENOR_DIAGONAL ETIQUETAS MAYOR
;

etiqueta ::= MENOR ETIQUETA VALOR IGUAL COMILLA PALABRA COMILLA MAYOR_DIAGONAL
        | MENOR ETIQUETA VALOR IGUAL COMILLA PALABRA COMILLA MAYOR_DIAGONAL
etiqueta
        ;









/* ------------------ Borrar Sitio Web ------------------- */
inicioBorrarSitio ::= MENOR PARAMETROS MAYOR id4 MENOR_DIAGONAL PARAMETROS MAYOR
;

/* ------------------ Borrar Pagina Web ------------------- */
inicioBorrarPagina ::= MENOR PARAMETROS MAYOR id4 MENOR_DIAGONAL PARAMETROS MAYOR
;

/* ------------------ Parte en común entre Borrar Sitio y Borrar Pagina ---------
--------- */
id4 ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA ID COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR
;

/* ------------------ Borrar Componente ------------------- */
inicioBorrarComponente ::= MENOR PARAMETROS MAYOR id5 MENOR_DIAGONAL PARAMETROS
MAYOR
;

id5 ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA ID COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR pagina1
;

pagina1 ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA PAGINA COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR
;
```

```
/* ------------------ Agregar y Modificar el Componente ------------------ */
inicioAgregarOModificarComponente ::= MENOR PARAMETROS MAYOR id6
;

id6 ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA ID COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR pagina2
;

pagina2 ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA PAGINA COMILLA MAYOR contexto
MENOR_DIAGONAL PARAMETRO MAYOR clase
;

clase ::= MENOR PARAMETRO NOMBRE IGUAL COMILLA CLASE COMILLA MAYOR tipo
;

tipo ::= CORCHETE_I TITULO CORCHETE_D MENOR_DIAGONAL PARAMETRO MAYOR
MENOR_DIAGONAL PARAMETROS MAYOR atributos1
        | CORCHETE_I PARRAFO CORCHETE_D MENOR_DIAGONAL PARAMETRO MAYOR
MENOR_DIAGONAL PARAMETROS MAYOR atributos1
        | CORCHETE_I IMAGEN CORCHETE_D MENOR_DIAGONAL PARAMETRO MAYOR
MENOR_DIAGONAL PARAMETROS MAYOR atributos2
        | CORCHETE_I VIDEO CORCHETE_D MENOR_DIAGONAL PARAMETRO MAYOR
MENOR_DIAGONAL PARAMETROS MAYOR atributos3
        | CORCHETE_I MENU CORCHETE_D
        ;

atributos1 ::= MENOR ATRIBUTOS MAYOR atributo1 MENOR_DIAGONAL ATRIBUTOS MAYOR
;

atributo1 ::= MENOR ATRIBUTO NOMBRE IGUAL COMILLA TEXTO COMILLA MAYOR contenido
MENOR_DIAGONAL ATRIBUTO MAYOR
        | MENOR ATRIBUTO NOMBRE IGUAL COMILLA TEXTO COMILLA MAYOR contenido
MENOR_DIAGONAL ATRIBUTO MAYOR MENOR ATRIBUTO NOMBRE IGUAL COMILLA ALINEACION
atributo2
        | MENOR ATRIBUTO NOMBRE IGUAL COMILLA TEXTO COMILLA MAYOR contenido
MENOR_DIAGONAL ATRIBUTO MAYOR MENOR ATRIBUTO NOMBRE IGUAL COMILLA COLOR atributo3
        ;
```

```
atributo2 ::= COMILLA MAYOR CORCHETE_I IZQUIERDA CORCHETE_D MENOR_DIAGONAL
ATRIBUTO MAYOR
        | COMILLA MAYOR CORCHETE_I CENTRAR CORCHETE_D MENOR_DIAGONAL ATRIBUTO
MAYOR
        | COMILLA MAYOR CORCHETE_I DERECHA CORCHETE_D MENOR_DIAGONAL ATRIBUTO
MAYOR
        | COMILLA MAYOR CORCHETE_I IZQUIERDA CORCHETE_D MENOR_DIAGONAL ATRIBUTO
MAYOR MENOR ATRIBUTO NOMBRE IGUAL COMILLA COLOR atributo3
        | COMILLA MAYOR CORCHETE_I CENTRAR CORCHETE_D MENOR_DIAGONAL ATRIBUTO
MAYOR MENOR ATRIBUTO NOMBRE IGUAL COMILLA COLOR atributo3
        | COMILLA MAYOR CORCHETE_I DERECHA CORCHETE_D MENOR_DIAGONAL ATRIBUTO
MAYOR MENOR ATRIBUTO NOMBRE IGUAL COMILLA COLOR atributo3
        ;

atributo3 ::= COMILLA MAYOR color MENOR_DIAGONAL ATRIBUTO MAYOR
;

atributos2 ::= MENOR ATRIBUTOS MAYOR atributo4 MENOR_DIAGONAL ATRIBUTOS MAYOR
;

atributo4 ::= MENOR ATRIBUTO NOMBRE IGUAL COMILLA ORIGEN COMILLA MAYOR direccion
MENOR_DIAGONAL ATRIBUTO MAYOR
        | MENOR ATRIBUTO NOMBRE IGUAL COMILLA ORIGEN COMILLA MAYOR direccion
MENOR_DIAGONAL ATRIBUTO MAYOR MENOR ATRIBUTO NOMBRE IGUAL COMILLA ALINEACION
atributo5
        | MENOR ATRIBUTO NOMBRE IGUAL COMILLA ORIGEN COMILLA MAYOR direccion
MENOR_DIAGONAL ATRIBUTO MAYOR MENOR ATRIBUTO NOMBRE IGUAL COMILLA ALTURA
atributo6
        ;

atributo5 ::= COMILLA MAYOR CORCHETE_I IZQUIERDA CORCHETE_D MENOR_DIAGONAL
ATRIBUTO MAYOR MENOR ATRIBUTO NOMBRE IGUAL COMILLA ALTURA atributo6
        | COMILLA MAYOR CORCHETE_I CENTRAR CORCHETE_D MENOR_DIAGONAL ATRIBUTO
MAYOR MENOR ATRIBUTO NOMBRE IGUAL COMILLA ALTURA atributo6
        | COMILLA MAYOR CORCHETE_I DERECHA CORCHETE_D MENOR_DIAGONAL ATRIBUTO
MAYOR MENOR ATRIBUTO NOMBRE IGUAL COMILLA ALTURA atributo6
        ;

atributos3 ::= MENOR ATRIBUTOS MAYOR atributo8 MENOR_DIAGONAL ATRIBUTOS MAYOR
;

atributo8 ::= MENOR ATRIBUTO NOMBRE IGUAL COMILLA ORIGEN COMILLA MAYOR direccion
MENOR_DIAGONAL ATRIBUTO MAYOR MENOR ATRIBUTO NOMBRE IGUAL COMILLA ALTURA
atributo6
        ;
```

```
/* Complemento de Imagen y Video */
atributo6 ::= COMILLA MAYOR numero MENOR_DIAGONAL ATRIBUTO MAYOR MENOR ATRIBUTO
NOMBRE IGUAL COMILLA ANCHO atributo7
        ;

atributo7 ::= COMILLA MAYOR numero MENOR_DIAGONAL ATRIBUTO MAYOR
;









/* ------------------ Texto ------------------ */
contexto ::= CORCHETE_I PALABRA CORCHETE_D
;

/* ------------------ Texto ------------------ */
contenido ::= CORCHETE_I CONTENIDO CORCHETE_D
;

/* ------------------ Url ------------------ */
direccion ::= CORCHETE_I URL CORCHETE_D
;

/* ------------------ Fecha ------------------ */
fecha ::= CORCHETE_I FECHA CORCHETE_D
;

/* ------------------ Numero ------------------ */
numero ::= CORCHETE_I NUMERO CORCHETE_D
;

/* ------------------ Color ------------------ */
color ::= CORCHETE_I NUMERO_COLOR CORCHETE_D
;
```

Gramática Consultas

Para la verificación de las consultas hicimos uso de otro analizador para que fuera más fácil el manejo de la información. Como primer paso definimos nuestro analizador léxico utilizando jflex, como primero paso pusimos información personal de la clase como es la ubicación donde queremos que se guarde la clase, luego definimos elementos que utilizaremos como lo es cup, la línea y la columna. Por ultimo definimos cada uno de los tokens que deseamos retornar cuando se cumpla con ciertas características.

```
package com.mycompany.aplicacioncliente.main.analizadoresconsultas;

import java_cup.runtime.*;

%% // Separador de Area

%class LexicoConsultas
%public
%cup
%line
%column
%eofval{
    return symbol(sym.EOF);
%eofval}

%{
    private Symbol symbol(int type) {
        return new Symbol(type, yyline+1, yycolumn+1);
    }

    private Symbol symbol(int type, Object value) {
        return new Symbol(type, yyline+1, yycolumn+1, value);
    }
%}

Consultar = [cC][oO][nN][sS][uU][lL][tT][aA][rR]
Sitio = [vV][iI][sS][iI][tT][aA][sS][_][sS][iI][tT][iI][oO]
Pagina = [vV][iI][sS][iI][tT][aA][sS][_][pP][aA][gG][iI][nN][aA]
Populares = [pP][aA][gG][iI][nN][aA][sS][_][pP][oO][pP][uU][lL][aA][rR][eE][sS]
Componente = [cC][oO][mM][pP][oO][nN][eE][nN][tT][eE]
Todos = [tT][oO][dD][oO][sS]
Titulo = [tT][iI][tT][uU][lL][oO]
Parrafo = [pP][aA][rR][rR][aA][fF][oO]
Imagen = [iI][mM][aA][gG][eE][nN]
Video = [vV][iI][dD][eE][oO]
Menu = [mM][eE][nN][uU]
```

```
Palabra = [[_]|[-]|[$]][[a-zA-Z]|[0-9]][[a-zA-Z][0-9]]*
Saltos = \r|\n|\r\n
Espacios = {Saltos} | [ \t\f]

%% // Separador de Area

<YYINITIAL> {

    "\""                    { return symbol(sym.COMILLA, yytext()); }

    ","                     { return symbol(sym.COMA, yytext()); }

    "."                     { return symbol(sym.PUNTO, yytext()); }

    ";"                     { return symbol(sym.PUNTO_Y_COMA, yytext()); }

    {Consultar}             { return symbol(sym.CONSULTAR, yytext()); }

    {Sitio}                 { return symbol(sym.VISITAS_SITIO, yytext()); }

    {Pagina}                { return symbol(sym.VISITAS_PAGINA, yytext()); }

    {Populares}             { return symbol(sym.PAGINAS_POPULARES, yytext()); }

    {Componente}            { return symbol(sym.COMPONENTE, yytext()); }

    {Todos}                 { return symbol(sym.TODOS, yytext()); }

    {Titulo}                { return symbol(sym.TITULO, yytext()); }

    {Parrafo}               { return symbol(sym.PARRAFO, yytext()); }

    {Imagen}                { return symbol(sym.IMAGEN, yytext()); }

    {Video}                 { return symbol(sym.VIDEO, yytext()); }

    {Menu}                  { return symbol(sym.MENU, yytext()); }

    {Palabra}               { return symbol(sym.PALABRA, yytext()); }

    {Espacios}              {/* ignoramos */}
}

[^]                         { throw new Error("Error Léxico caracter Invalido en la
linea " + (yyline+1) + ", columna " + (yycolumn+1) + ": " + yytext());
```

El siguiente elemento es el analizador sintáctico donde nuevamente haremos uso cup para su realización como primer punto volvemos a definir los métodos que nos van a servir para retornar información importante al usuario y notificarle de cualquier error que se presente durante el análisis del contenido del panel, como siguiente punto se procedió a determinar el terminales y no terminales con el cuales íbamos estar trabajando las producciones y para el caso de las producciones se buscó que pudiéramos cumplir con cada uno de los casos que se nos presentarán.

```
package com.mycompany.aplicacioncliente.main.analizadoresconsultas;

import java_cup.runtime.*;

parser code {:

    private Symbol symbol;
    private String contenido;

    public SintacticoConsultas(LexicoConsultas lexico) {
        super(lexico);
    }

    public void syntax_error(Symbol cur_token) {
        symbol = cur_token;
        contenido = (String)(cur_token.value);
    }

    public Symbol getSymbol() {
        return this.symbol;
    }

    public String getContenido() {
        return this.contenido;
    }
:}

/* Terminales */
terminal            CONSULTAR, VISITAS_SITIO, VISITAS_PAGINA, PAGINAS_POPULARES,
COMPONENTE, COMILLA, COMA, PUNTO, PUNTO_Y_COMA, TODOS, TITULO, PARRAFO, IMAGEN,
VIDEO, MENU;
terminal String     PALABRA;

/* No Terminales */
non terminal        inicio, componente, id1, id2;

/* Gramatica */
```

```
start with inicio;

inicio ::= CONSULTAR VISITAS_SITIO id1
        | CONSULTAR VISITAS_PAGINA id2
        | CONSULTAR PAGINAS_POPULARES id1
        | CONSULTAR COMPONENTE componente
        ;

componente ::= TODOS id2
        | TITULO id2
        | PARRAFO id2
        | IMAGEN id2
        | VIDEO id2
        | MENU id2
        ;

id1 ::= COMILLA PALABRA COMILLA COMA id1
        | COMILLA PALABRA PUNTO PALABRA COMILLA PUNTO_Y_COMA
        ;

id2 ::= COMILLA PALABRA PUNTO PALABRA COMILLA COMA id2
        | COMILLA PALABRA PUNTO PALABRA COMILLA PUNTO_Y_COMA
        ;
```