Capitulo V Implementación

5.1 Situación Actual

Como ya se mencionaba en el primer capitulo, ya existe un sintetizador que ha sido desarrollado con anterioridad, daremos una breve introducción del trabajo ya realizado.

Este sintetizador trabaja con un solo corpus de voz, el cual esta pregrabado, este a su vez, era almacenado en memoria, como ya sabemos el acceso a memoria es lo mas caro, y para acceder a el, se realizaban una búsqueda secuencial, lo cual implica mucho tiempo para el resultado solicitado. Esto causaba un impacto demasiado fuerte en cuanto al desempeño, incluyendo el tiempo de acceso a la búsqueda y el acceso al corpus.

Como lo mencionábamos antes, la búsqueda realizada era lineal sobre todo el corpus y la cual generaba una selección sobre las palabras, y se agregaban mediante una lista, la cual se generaba con la carga de todo el corpus a memoria y la búsqueda se convertía en secuencial, es decir buscaba uno por uno, palabras y fonemas, así es como se realizaba el proceso de búsqueda en el trabajo anterior.

Analizando la situación actual, lo que se propone, es crear una base de datos, ya que así podremos utilizar todas las ventajas de las bases de datos para almacenar y manipular más de 3000 archivos, entre fonemas, palabras, frases, ya que nosotros manejaremos otro corpus mas, esto nos beneficiara en cuanto al tiempo de ejecución y de búsqueda dependiendo de la palabra o frase a sintetizar.

El desarrollo de la base de datos empieza por crear las tablas, las cuales contendrán la información de cada una de las palabras, frases y fonemas que existen en nuestro corpus de voz.

5.2 Importación del Corpus

También proponemos una aplicación en Visual Basic, la cual nos ayudara en un futuro para cargar diferentes tipos de corpus, esto es, no tendremos que modificar absolutamente nada de código para que estos corpus sean incorporados a la base de datos.

```
Option Explicit
Private Sub cmdImportar_Click()
frmImportar.Show
Unload Me
End Sub
Private Sub cmdSalir_Click()
Unload Me
End Sub
```

Con este frame, al seleccionar la opción de importar nos mandara al siguiente frame donde se importaran los corpus.

```
Option Explicit
Private Sub cmdCerrar_Click()
   Unload Me
End Sub
Private Sub cmdImportar_Click()
   Dim intI As Integer, intFile As Integer, intCorpus As Integer, intJ As Integer
   Dim intPos As Integer, intPos2 As Integer
   Dim strWRD As String, strPHN As String, strTXT As String
   Dim strFile As String, strFileName As String, strCad As String
   Dim strInfo As String, strInicio As String, strFin As String
   Dim blnWRD As Boolean, blnPHN As Boolean, blnTXT As Boolean
   Dim cnCorpus As ADODB.Connection, rsCorpus As ADODB.Recordset
```

```
    Set cnCorpus = New ADODB.Connection
    Set rsCorpus = New ADODB.Recordset
    'Add Corpus
    cnCorpus.Open "DSN=Tesis"
    rsCorpus.Open "Select * FROM corpus ORDER BY corpusid", cnCorpus, adOpenDynamic,
adLockOptimistic
    If rsCorpus.EOF Then
       intCorpus = 1
    Else
       rsCorpus.MoveLast
       intCorpus = rsCorpus("corpusid") + 1
    End If
    rsCorpus.AddNew
    rsCorpus("corpusid") = intCorpus
    rsCorpus("nombre") = txtNombre.Text
    rsCorpus("ruta") = txtWAV.Text
    rsCorpus("Prefijo") = txtPrefijo.Text
    rsCorpus.Update
    rsCorpus.Close
    For intI = Val(txtInicio.Text) To Val(txtFin.Text)
       blnWRD = False
       blnPHN = False
       blnTXT = False
       strFile = txtPrefijo.Text & CStr(Format(intI, "0000"))
       If Dir(txtRuta.Text & strFile & ".wrd") <> "" Then blnWRD = True
       If Dir(txtRuta.Text & strFile & ".phn") <> "" Then blnPHN = True
       If Dir(txtRuta.Text & strFile & ".txt") <> "" Then blnTXT = True
       If blnWRD And blnPHN And blnTXT Then
          'Add Frase
          strFileName = txtRuta.Text & strFile & ".txt"
          intFile = FreeFile
          Open strFileName For Input As intFile
          Line Input #intFile, strCad
          Close intFile
          rsCorpus.Open "frase", cnCorpus, adOpenDynamic, adLockOptimistic, adCmdTable
          rsCorpus.AddNew
          rsCorpus("corpusid") = intCorpus
          rsCorpus("fraseid") = intI
```

```
rsCorpus("frase") = strCad
rsCorpus.Update
rsCorpus.Close
'Add Palabras
strFileName = txtRuta.Text & strFile & ".wrd"
intFile = FreeFile
Open strFileName For Input As intFile
Line Input #intFile, strCad
Line Input #intFile, strCad
intJ = 1
rsCorpus.Open "Palabra", cnCorpus, adOpenDynamic, adLockOptimistic, adCmdTable
Do While Not EOF(intFile)
   Line Input #intFile, strCad
   intPos = InStr(strCad, " ")
   strInicio = Left(strCad, intPos)
   intPos2 = InStr(intPos + 1, strCad, " ")
   strFin = Mid(strCad, intPos + 1, intPos2 - intPos)
   strInfo = Right(strCad, Len(strCad) - intPos2)
   rsCorpus.AddNew
   rsCorpus("corpusid") = intCorpus
   rsCorpus("fraseid") = intI
   rsCorpus("palabraid") = intJ
   rsCorpus("palabra") = Trim(strInfo)
   rsCorpus("inicio") = Val(Trim(strInicio))
   rsCorpus("fin") = Val(Trim(strFin))
   rsCorpus.Update
   intJ = intJ + 1
Loop
Close intFile
rsCorpus.Close
'Add Fonemas
strFileName = txtRuta.Text & strFile & ".phn"
intFile = FreeFile
Open strFileName For Input As intFile
Line Input #intFile, strCad
Line Input #intFile, strCad
intJ = 1
```

```
            rsCorpus.Open "Fonema", cnCorpus, adOpenDynamic, adLockOptimistic, adCmdTable
        Do While Not EOF(intFile)
            Line Input #intFile, strCad
            intPos = InStr(strCad, " ")
            strInicio = Left(strCad, intPos)
            intPos2 = InStr(intPos + 1, strCad, " ")
            strFin = Mid(strCad, intPos + 1, intPos2 - intPos)
            strInfo = Right(strCad, Len(strCad) - intPos2)
            rsCorpus.AddNew
            rsCorpus("corpusid") = intCorpus
            rsCorpus("fraseid") = intI
            rsCorpus("fonemaid") = intJ
            rsCorpus("fonema") = Trim(strInfo)
            rsCorpus("inicio") = Val(Trim(strInicio))
            rsCorpus("fin") = Val(Trim(strFin))
            rsCorpus.Update
            intJ = intJ + 1
        Loop
        Close intFile
        rsCorpus.Close
      End If
      pgbProgress.Value = intI / Val(txtFin.Text) * 100
   Next
   MsgBox "Importacion Exitosa!!", vbOKOnly, "Importacion"
End Sub
Private Sub Form_Unload(Cancel As Integer)
   frmMain.Show
   Unload Me
End Sub
```

Aquí es donde proporcionaremos la ruta donde se encuentran nuestros corpus dentro de la computadora, para que la aplicación los importe directamente a la base de datos. Los numero que aparecen en el campo de prefijo, se refieren del corpus a importar, es decir su primer elemento empieza en el numero uno y el ultimo termina en el 1500, todos estos archivos son importados a las tablas de la base de datos que a continuación explicaremos.

A continuación, mostraremos como es que se crearon las tablas y su implementación después de haber instalado MySQL, mencionada en el capitulo anterior.

5.3 Implementación de las Tablas de la Base de Datos

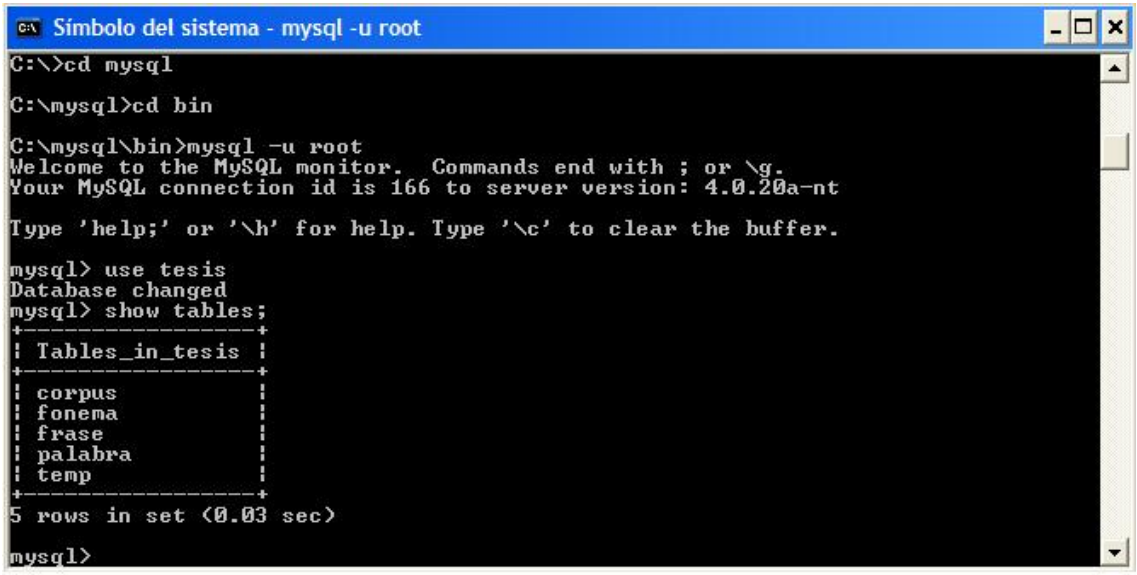

Fig. 5.1 Creación de las Tablas del corpus

En esta figura se despliegan las tablas que fueron creadas para guardar la información del corpus, como lo son los mismos corpus, es decir sus nombres, los fonemas, las frases, las palabras y una tabla Temp, que se explicara más adelante.

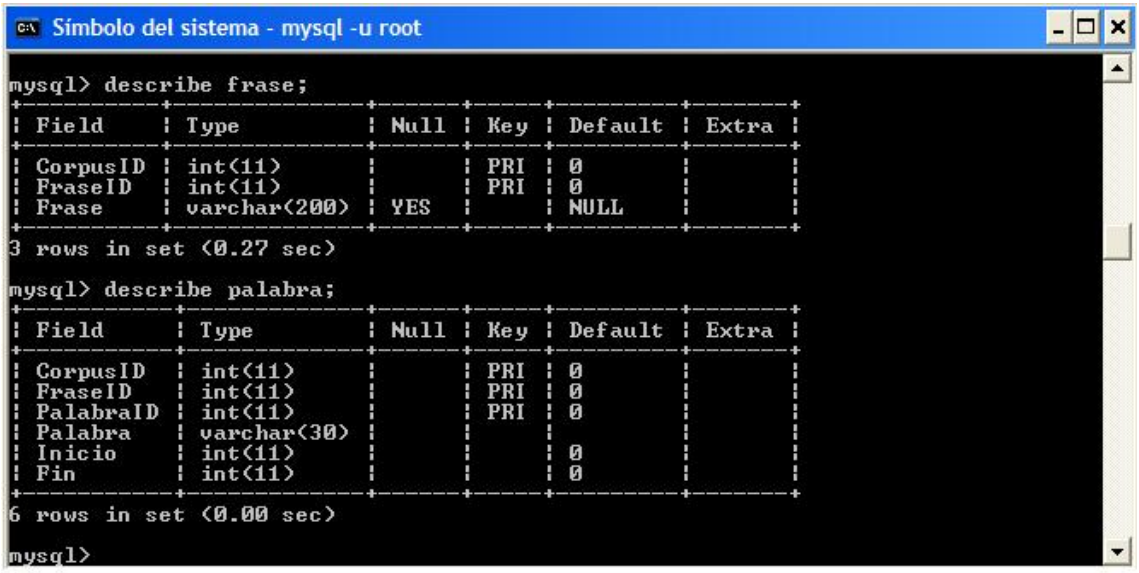

Fig. 5.2. Descripción del contenido de las tablas Frase y Palabra

Estas tablas nos muestran el contenido de cada una de ellas, en el caso de la tabla frase, su contenido es el siguiente, CorpusID, FraseID, y frase, cada una de ellas, la información necesaria para ser localizadas por nuestro sistema. Dentro de la tabla de Palabra, encontraremos, como en la tabla anterior información del Corpus de la frase y de la palabra, además del inicio y del fin de cada palabra, el inicio se refiere a los milisegundos en los cuales principia la palabra, y así a su vez el fin donde termina.

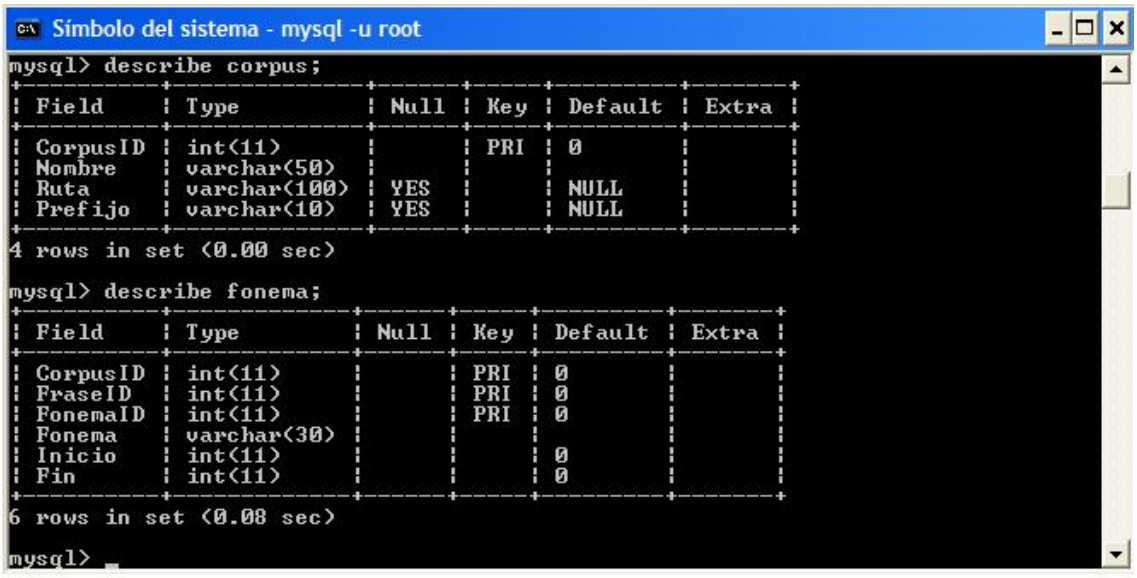

Fig. 5.3 Descripción de las tablas Corpus y Fonema

En estas tablas encontramos, la información del corpus, cuales son los que tenemos cargados en la base de datos, la ruta, que es donde el sistema los encontrara, un prefijo, que es el nombre del corpus a usar.

En el caso de fonema encontramos, la misma información esto es para identificar en que lugar se encuentran los fonemas, además del fonema como tal y su inicio y su fin.

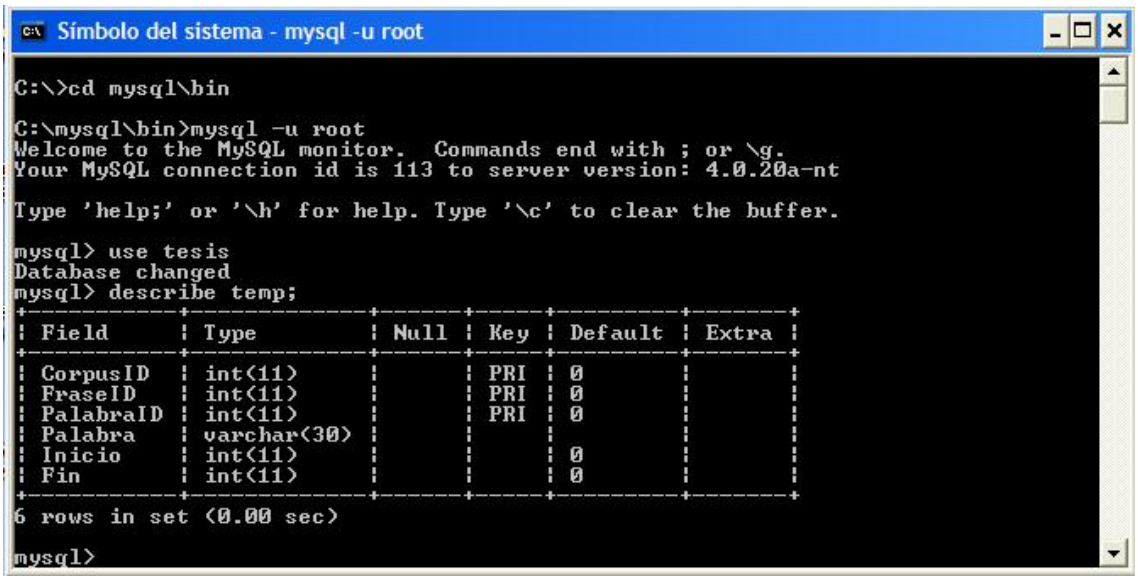

Fig. 5.4 Descripción de la tabla Temp.

Esta tabla es creada con la información de cada una de las palabras, frases, su inicio y fin, como en las anteriores, solo que esta tabla se crea para guardar los candidatos que serán concatenados para la síntesis.

Una vez que hemos creado las tablas y que ya tenemos nuestros corpus importados en la base de datos, veamos como es que realizamos el código.
La implementación del código como lo mencionamos con anterioridad, tuvo un proceso de limpieza y de ahí empezamos a configurar nuestro sistema.

En primer lugar implementamos un nuevo código para poder modificar nuestro frame, que es ahí donde los usuarios proporcionaran, el texto para ser sintetizado.

5.4 UnitSelection

El frame es la interfaz gráfica que interactúa con el usuario. En esta parte es donde el usuario proporciona la información a sintetizar, es decir una frase, oración o palabra, al ingresar el texto es necesario seleccionar un corpus para que la información pueda ser sintetizada, este esta cargado dentro de la clase de unitselection y se genera de la siguiente manera:

```
public unitselection() {
    super("Tlatoa - Memo Velez");
    CorpusDb corpus;
    ArrayList arrCorpus;
    setSize(500,500);
    setBackground(Color.gray);
    m_btnIniciar = new JButton("Leer");
    m_btnIniciar.addActionListener(this);
    m_btnLimpiar = new JButton("Limpiar");
    m_btnLimpiar.addActionListener(this);
    m_txtTexto = new JTextArea();
```

```
    m_cboCorpus = new JComboBox();
    getContentPane().setLayout(null);
    getContentPane().add(m_txtTexto);
    m_txtTexto.setBounds(10,150,450,400);
    getContentPane().add(m_btnIniciar);
    m_btnIniciar.setBounds(480,500,100,30);
    getContentPane().add(m_btnLimpiar);
    m_btnLimpiar.setBounds(480,450,100,30);
    getContentPane().add(m_cboCorpus);
    m_cboCorpus.setBounds(10,100,400,30);
    resize(600,600);
    corpus=new CorpusDb();
    arrCorpus=corpus.loadAllCorpus();
    m_cboCorpus.addItem("");
    for(int i=0;i<arrCorpus.size();i++)
       m_cboCorpus.addItem(arrCorpus.get(i));
    m_iCorpus=-1;
    //m_strTexto="Esta es una prueba";
    //recognize();
  }
```

5.4.1 Recognize

Recognize es un método que se encuentra dentro de nuestra clase principal, el cual tiene la función de reconocer el corpus que se va a usar para la creación de la síntesis, este se ejecuta cuando el usuario selecciona el botón de leer.

```
private void recognize(){
    CorpusDb corpus;
    String strCorpus;
    creawav("del wavs\\*.* /q");
    corpus=new CorpusDb();
    strCorpus=m_cboCorpus.getItemAt(m_cboCorpus.getSelectedIndex()).toString();
```

```
        //strCorpus="timo";
        m_iCorpus=corpus.getCorpusID(strCorpus);
        m_strTexto = m_txtTexto.getText();
        procesandoTexto();
        concatenaWav();
        tocaAudio();
        //toca();
        Duerme duerme = new Duerme();
    }
```

5.5 Constructor de Oraciones

En esta clase, se almacena por medio de un vector la información que va a ser sintetizada.

```
Public Class ConstructorDeOraciones {
        Vector  vInfoSintetizar ;
        unitselection unit;
        public ConstructorDeOraciones(Vector por_sintetizar,unitselction theunit){
                vInfoSintetizar= por_sintetizar;
                unit = theunit;
        }
```

5.5.1 Localiza Candidatos

El método localiza candidatos es el encargado de localizar por medio de una lista ligada, los elementos que sean iguales a la palabra proporcionada para ser sintetizada, así como los contextos que esta contiene, es decir, el contexto izquierdo y derecho, así como el de recorrer el vector de las listas ligadas para seleccionar el mejor candidato a usar, el método se encuentra dentro de la clase Constructor de Oraciones.

```
for (int i=0;i<vInfoSintetizar.size();i++){
        contexto_cen = (String) vInfoSintetizar.get(i);
        if(i!=0)
           contexto_izq = (String) vInfoSintetizar.get(i-1);
        if(i!= (vInfoSintetizar.size()-1))
           contexto_der = (String) vInfoSintetizar.get(i+1);
        //recorrido para todo el vector de listas ligadas
        blnAtras=false;
        blnAdelante=false;
        for (int j=0;j<lista.getSize();j++){
           //System.out.println(lista.getAt(j).getFrase());
           strCentro=lista.getAt(j).getCentral().getPalabra();
           strIzquierda=lista.getAt(j).getAnterior().getPalabra();
           strDerecha=lista.getAt(j).getSiguiente().getPalabra();

                if(contexto_cen.compareToIgnoreCase(lista.getAt(j).getCentral().getPala
                bra())=   =0){

                if(contexto_izq.compareToIgnoreCase(lista.getAt(j).getAnterior().getPal
                abra())==0){
                 blnAtras=true;
              }

                if(contexto_der.compareToIgnoreCase(lista.getAt(j).getSiguiente().getPa
                labra())==0){
                 blnAdelante=true;
              }
                if (blnAtras && blnAdelante){
                 iPeso=3;

                fron_izq=newFronteras(contexto_izq,lista.getAt(j).getAnterior().getInici
                o(),lista.getAt(j).getAnterior().getFin());
```

```
        fron_cen=newFronteras(contexto_cen,lista.getAt(j).getCentral().getInicio(),lista.
        getAt(j).getCentral().getFin());

        fron_der=newFronteras(contexto_der,lista.getAt(j).getSiguiente().getInicio(),list
        a.getAt(j).getSiguiente().getFin());

        candidato=newCandidatosConcatenacion(lista.getAt(j).getWAV(),fron_izq,fron
        _cen,fron_der,iPeso,i,"111");
       break;
}   else{
      if (blnAtras || blnAdelante){
          iPeso=2;
          if (blnAtras){

                fron_izq=newFronteras(contexto_izq,lista.getAt(j).getAnterior().getInici
                o(),lista.getAt(j).getAnterior().getFin());

                fron_cen=newFronteras(contexto_cen,lista.getAt(j).getCentral().getInicio
                (),lista.getAt(j).getCentral().getFin());
               fron_der = new Fronteras();

                candidato=newCandidatosConcatenacion(lista.getAt(j).getWAV(),fron_i
                zq,fron_cen,fron_der,iPeso,i,"110");
         }else{
                    fron_izq = new Fronteras();

                fron_cen=newFronteras(contexto_cen,lista.getAt(j).getCentral().getInicio
                (),lista.getAt(j).getCentral().getFin());

                fron_der=newFronteras(contexto_der,lista.getAt(j).getSiguiente().getInic
                io(),lista.getAt(j).getSiguiente().getFin());

                candidato=newCandidatosConcatenacion(lista.getAt(j).getWAV(),fron_i
                zq,fron_cen,fron_der,iPeso,i,"011");
                   }
        }else{
                iPeso=1;
                fron_izq = new Fronteras();

                fron_cen=newFronteras(contexto_cen,lista.getAt(j).getCentral().getInicio
                (),lista.getAt(j).getCentral().getFin());
                fron_der = new Fronteras();

                candidato=newCandidatosConcatenacion(lista.getAt(j).getWAV(),fron_i
                zq,fron_cen,fron_der,iPeso,i,"010");
         }
   }
        CandidatosConcatenacion elige_candidato = (CandidatosConcatenacion)
        contextos.get(i);
```

```
            if(elige_candidato.peso < candidato.peso){
               contextos.removeElementAt(i);
               contextos.add(i,(CandidatosConcatenacion) candidato);
            }//fin del if de elige candidato
         }
      }
```

5.6 Corpus DB

La Clase Corpus DB es la encargada de comunicar nuestra base de Datos con el sistema, aquí encontraremos la conexión por medio del driver que se necesita para realizar la conexión con la base de datos y del corpus con JAVA.

```
public CorpusDb(){
    m_strClass="org.gjt.mm.mysql.Driver";
    m_strURL="jdbc:mysql://localhost/Tesis";
  }
```

Una vez que se haya hecho la conexión con la base de datos, se manda llamar a un método llamado LOADALLCORPUS, el cual ejecuta la conexión con el driver de mysql, y se crea un estado para ejecutar el Query, este Query es para seleccionar el corpus dentro de este método llamado LOAD CORPUS donde se seleccionará el corpus deseado.

```
public ArrayList loadAllCorpus(){
    try{
       ArrayList arrCorpus;
       Class.forName(m_strClass);
       m_Con=DriverManager.getConnection(m_strURL,"root","");
       m_Stmt = m_Con.createStatement();
```

```
            ResultSet rs = m_Stmt.executeQuery("select  *  from corpus;");
            arrCorpus=new ArrayList();
            while (rs.next()){
               arrCorpus.add(rs.getString("Nombre"));
            }
            m_Stmt.close();
            m_Con.close();
            return arrCorpus;
         }
         catch( Exception e) {
            e.printStackTrace( );
         }
         return null;
      }

public ArrayList loadCorpus(int iCorpus){
         try{
            ArrayList arrCorpus;
            String strSQL;
            Class.forName(m_strClass);
            m_Con=DriverManager.getConnection(m_strURL,"root","");
            m_Stmt = m_Con.createStatement();
            strSQL="select  *  from corpus where CorpusID=" + iCorpus;
            ResultSet rs = m_Stmt.executeQuery(strSQL);
            if (rs.next()){
               arrCorpus=new ArrayList();
               arrCorpus.add(rs.getString("CorpusID"));
               arrCorpus.add(rs.getString("Nombre"));
               arrCorpus.add(rs.getString("Ruta"));
               arrCorpus.add(rs.getString("Prefijo"));
            }else
               return null;
            m_Stmt.close();
            m_Con.close();
```

```
        return arrCorpus;
    }
    catch( Exception e) {
        e.printStackTrace( );
    }
    return null;
}
```

Ya que se haya seleccionado el corpus a usar, se llama otro método el cual crea una lista de candidatos en la tabla Temp, antes mencionada, aquí se genera una lista de puros candidatos, y se ejecuta un query para ingresar el Id del corpus, de la palabra su inicio y su fin, así miso para la frase.

```
public ListaCandidatos getCandidatos(Vector vInfo,int iCorpus){
    int iFrase;
    String strSQL,strTemp,strWAV;
    ArrayList arrCorpus;
    PalabraContigua palIzquierda,palCentral,palDerecha;
    ListaCandidatos lista;
    NodoCandidato nodo;
    arrCorpus=loadCorpus(iCorpus);
    strTemp="";
    strTemp=(String)strTemp.valueOf(iCorpus);
    try{
        Class.forName(m_strClass);
        m_Con=DriverManager.getConnection(m_strURL,"root","");
        m_Stmt = m_Con.createStatement();
        strSQL="DELETE FROM temp";
        m_Stmt.executeQuery(strSQL);
        m_Stmt.close();
        m_Stmt = m_Con.createStatement();
        strSQL="INSERT INTO temp (CorpusID, FraseID, PalabraID, Palabra,
        Inicio,Fin) SELECT CorpusID, FraseID, PalabraID, Palabra, ";
```

```
strSQL+="Inicio,Fin FROM palabra WHERE (CorpusID=" + strTemp + ") AND (";
for (int i=0;i<vInfo.size();i++){
   strSQL+="(Palabra='" + vInfo.get(i).toString() + "')";
   if (i<vInfo.size()-1)
      strSQL+=" OR ";
}

m_Stmt.executeQuery(strSQL);
m_Stmt.close();
strSQL="SELECT CorpusID, FraseID, PalabraID, Palabra, Inicio, Fin, 0 AS Pos FROM temp ";
strSQL+="UNION SELECT temp.CorpusID, temp.FraseID, temp.PalabraID-1 AS PalabraID, palabra.Palabra, palabra.Inicio, ";
strSQL+="palabra.Fin, 1 AS Pos FROM temp INNER JOIN palabra ON (temp.PalabraID-1=palabra.PalabraID) AND ";
strSQL+="(temp.FraseID=palabra.FraseID)AND(temp.CorpusID=palabra.CorpusID)";
strSQL+="UNION SELECT temp.CorpusID, temp.FraseID, temp.PalabraID+1 AS PalabraID, palabra.Palabra, palabra.Inicio, ";
strSQL+="palabra.Fin, 2 AS Pos FROM temp INNER JOIN palabra ON (temp.PalabraID+1 = palabra.PalabraID) AND ";
strSQL+="(temp.FraseID = palabra.FraseID) AND (temp.CorpusID = palabra.CorpusID) ORDER BY 1,2,7,3";
m_Stmt=m_Con.createStatement();
ResultSet rs=m_Stmt.executeQuery(strSQL);
lista=new ListaCandidatos();
while (rs.next()){
   palCentral=palIzquierda=palDerecha=null;
   iFrase=-1;
   if (rs.getInt("Pos")==0){
      iFrase=rs.getInt("FraseID");
      strTemp=(String)strTemp.valueOf(iFrase);
      while(strTemp.length()!=4)
```

```
                strTemp="0"+strTemp;
            strTemp=strTemp+".wav";
            strWAV=arrCorpus.get(2).toString()+arrCorpus.get(3).toString()+strTemp;
               palCentral=newPalabraContigua(rs.getInt("PalabraID"),
               rs.getString("Palabra"),rs.getInt("Inicio"),rs.getInt("Fin"));
               nodo=new NodoCandidato(palCentral,iFrase,strWAV);
                 lista.Insertar(nodo);


        }else
            if (rs.getInt("Pos")==1){
               NodoCandidatotemp=lista.find(rs.getInt("FraseID"),
               rs.getInt("PalabraID")+1);palIzquierda=newPalabraContigua(rs.getInt("P
               alabraID"), rs.getString("Palabra"),rs.getInt("Inicio"),rs.getInt("Fin"));
               temp.setAnterior(palIzquierda);
            }
               else{
               NodoCandidatotemp=lista.find(rs.getInt("FraseID"),
               rs.getInt("PalabraID")-1);
               palDerecha=newPalabraContigua(rs.getInt("PalabraID"),
               rs.getString("Palabra"),rs.getInt("Inicio"),rs.getInt("Fin"));
               temp.setSiguiente(palDerecha);
            }
      }
      m_Stmt.close();
      m_Con.close();
      return lista;
   }
   catch( Exception e) {
      e.printStackTrace( );
   }
   return null;
}
```

Después de ser ejecutado este query para la selección del corpus, se ejecuta uno mas, donde se hace la búsqueda dentro de la lista de candidatos, después de ser ejecutado se crea una nueva lista, para las palabras, regresando el ID de las mismas.

En el caso de los fonemas, el método que realiza la selección es getCandidatosFonema, que es un vector, el cual contiene, la información de los fonemas, así como sus wavs, y al igual que para las palabras, ejecuta un query para hacer la búsqueda de fonemas.

```
public Vector getCandidatosFonema(Vector vFonemasSilaba,int iCorpus){
int iFrase,iFraseAnt,iFonemaPos,iFonemaPosAnt,iLista,iFonCandidato,iFonemaID;

    String strSQL,strTemp,strWAV,strFonema;
    ArrayList arrCorpus;
    Vector vFonemasCandidato;
    FonemaCandidato fonCandidato;
    NodoFonema nodo;
    arrCorpus=loadCorpus(iCorpus);
    strTemp="";
    iFonemaPosAnt=0;
    iFraseAnt=0;
    iLista=-1;
    iFonCandidato=0;
    iFonemaID=0;
    strTemp=(String)strTemp.valueOf(iCorpus);
    vFonemasCandidato=new Vector(vFonemasSilaba.size());
    for (int i=0;i<vFonemasSilaba.size();i++){

      vFonemasCandidato.add(newFonemaCandidato(vFonemasSilaba.get(i).toString(
      )));
    }
    try{
      Class.forName(m_strClass);
      m_Con=DriverManager.getConnection(m_strURL,"root","");
      m_Stmt = m_Con.createStatement();
      strSQL="SELECT CorpusID, FraseID, FonemaID, Fonema, Inicio,
      Fin FROM fonema ";
      strSQL=strSQL+"Where (CorpusID=" + strTemp + ") AND (";
      for (int i=0;i<vFonemasSilaba.size();i++){
        strSQL+="(Fonema='" + vFonemasSilaba.get(i).toString() + "')";
        if (i<vFonemasSilaba.size()-1)
```

```
            strSQL+=" OR ";
        }
        strSQL+=") ORDER BY CorpusID, FraseID, FonemaID";
        ResultSet rs=m_Stmt.executeQuery(strSQL);
        while (rs.next()){
           iFrase=rs.getInt("FraseID");
           if (iFrase==1376)
              System.out.println("Tst");
           strTemp=(String)strTemp.valueOf(iFrase);
           while(strTemp.length()!=4)
              strTemp="0"+strTemp;
           strTemp=strTemp+".wav";

        strWAV=arrCorpus.get(2).toString()+arrCorpus.get(3).toString()+strTemp;
        strFonema=rs.getString("Fonema");

for(iFonemaPos=0;iFonemaPos<vFonemasSilaba.size();iFonemaPos++)

  if(strFonema.compareToIgnoreCase((String)vFonemasSilaba.get(iFonemaPos))==0)
break;

fonCandidato=(FonemaCandidato)vFonemasCandidato.get(iFonemaPos);
            if (iFrase==iFraseAnt && iFonemaPos==iFonemaPosAnt+1 &&
rs.getInt("FonemaID")==iFonemaID+1){//Misma Frase sig pos

fonCandidato=(FonemaCandidato)vFonemasCandidato.get(iFonCandidato);

nodo=newNodoFonema(strFonema,strWAV,rs.getInt("Inicio"),rs.getInt("Fin"));
               fonCandidato.getList(iLista).Insertar(nodo);
               if
                  (fonCandidato.getMaxListSize()<fonCandidato.getList(iLista).getCount(
        ))

                  fonCandidato.setMaxListSize(fonCandidato.getList(iLista).getCount());
           }
           else{
              if (iFonemaPos!=vFonemasSilaba.size()-1){//No es la ultima pos
                 iFonCandidato=iFonemaPos;
                 iLista=fonCandidato.addList();
                 nodo=new
                 NodoFonema(strFonema,strWAV,rs.getInt("Inicio"),rs.getInt("Fin"));
                 fonCandidato.getList(iLista).Insertar(nodo);
                 if
                 (fonCandidato.getMaxListSize()<fonCandidato.getList(iLista).getCount(
                 ))

                 fonCandidato.setMaxListSize(fonCandidato.getList(iLista).getCount());
              }
           else
              if(fonCandidato.getListSize()==0){ //Primer Elemento
```

```
                    iFonCandidato=iFonemaPos;
                    iLista=fonCandidato.addList();
                    nodo=new
                    NodoFonema(strFonema,strWAV,rs.getInt("Inicio"),rs.getInt("Fin"));
                    fonCandidato.getList(iLista).Insertar(nodo);
                  if
                    (fonCandidato.getMaxListSize()<fonCandidato.getList(iLista).getCount(
                    ))

                    fonCandidato.setMaxListSize(fonCandidato.getList(iLista).getCount());
                  }
               }
               iFraseAnt=iFrase;
               iFonemaPosAnt=iFonemaPos;
               iFonemaID=rs.getInt("FonemaID");
            }
            m_Stmt.close();
            m_Con.close();
            for(int i=0;i<vFonemasCandidato.size();i++)
               ((FonemaCandidato)vFonemasCandidato.get(i)).print();
            return vFonemasCandidato;
         }
         catch( Exception e) {
            e.printStackTrace( );
         }
         return null;
      }
}
```

5.7 Palabra Contigua

Esta es una estructura la cual contiene la información de una palabra

```
public PalabraContigua(int iPalabraID, String strPalabra,int iInicio,int iFin){
      m_iPalabraID=iPalabraID;
      m_strPalabra=strPalabra;
      m_iInicio=iInicio;
      m_iFin=iFin;
   }

   public String getPalabra(){
      return m_strPalabra;
   }

   public int getInicio(){
      return m_iInicio;
```

```
}

public int getFin(){
   return m_iFin;
}
```

5.8 Nodo Candidato

Nodo candidato es una clase, que realiza la función para guardar los posibles candidatos que serán concatenados, o sintetizados, quien toma la decisión de quien será el mejor candidato es el método de localiza candidatos como ya se vio anteriormente.

```
public NodoCandidato(){
      m_PalAnterior=m_PalCentral=m_PalSiguiente=null;
      m_iFrase=0;
      m_strWAV="";
      m_Sig=null;
   }

   public NodoCandidato(PalabraContigua PalAnterior ,PalabraContigua PalCentral,
PalabraContigua PalSiguiente,int iFrase,String strWAV){
      m_PalAnterior=PalAnterior;
      m_PalCentral=PalCentral;
      m_PalSiguiente=PalSiguiente;
      m_iFrase=iFrase;
      m_strWAV=strWAV;
      m_Sig=null;
   }

   public NodoCandidato(PalabraContigua PalCentral, int iFrase, String strWAV){
      m_PalAnterior=null;
```

```
        m_PalCentral=PalCentral;
        m_PalSiguiente=null;
        m_iFrase=iFrase;
        m_strWAV=strWAV;
        m_Sig=null;
    }
```

5.8.1 Lista Candidatos

La clase Lista Candidatos es la encargada de insertar los nodos que han sido creados con la información de los candidatos.

```
public ListaCandidatos() {
        m_Head=null;
        m_iTam=0;
    }

    public int getSize(){
        return m_iTam;
    }

    public void Insertar(NodoCandidato nodo){
        NodoCandidato temp;
        if (m_Head==null){
            m_Head=nodo;
            m_iTam=1;
            return;
        }
        for(temp=m_Head;temp.getSig()!=null;)
            temp=temp.getSig();
        temp.setSig(nodo);
        m_iTam++;
    }

    public NodoCandidato getAt(int index){
        int i;
        NodoCandidato temp;
        if (index>=m_iTam)
            return null;
        for(i=0,temp=m_Head;i<index;i++,temp=temp.getSig());
        return temp;
```

```
}

   public NodoCandidato find(int iFrase, int iPalabraID){
      for(NodoCandidato temp=m_Head;temp!=null;temp=temp.getSig()){
         if (temp.getFrase()==iFrase && temp.getCentral().getPalabraID()==iPalabraID)
            return temp;
      }
      return null;
   }
```

5.8.2 Fronteras

La clase fronteras nos proporciona la información de los wavs de cada palabra, aquí se sabe donde empieza y donde termina cada palabra, y es enviada a su vez a una clase que explicaremos en un paso mas adelante.

```
public class Fronteras
{
   long inf;
                long sup;
                String palabra;

                public Fronteras(){
                    palabra="hello";
                    inf=0;
                                sup=0;
                }

                public Fronteras(String palabra,long inf,long sup){
                     asignaPalabra(palabra);
                                        asignaInf(inf);
                                        asignaSup(sup);
                }

                /**
                *   Regresa un long que corresponde a el valor de la frontera inferior
                */
                public long regresaInf(){
                     return inf;
                }

                /**
                *   Regresa un long que corresponde a el valor de la frontera superior
```

```
        */


                public long regresaSup(){
                return sup;
        }


        /**
        *   Regresa un String que corresponde a el valor de la palabra
        */
        public String regresaPalabra(){
return palabra;
        }

        /**
        *   Como su nombre lo indica asigna un valor (long) a la frontera inferior
        */
        public void asignaInf(long inf){
                this.inf= inf;
        }

        /**
        *   Como su nombre lo indica asigna un valor (long) a la frontera superior
        */
        public void asignaSup(long sup){
             this.sup = sup;
        }


        /**
        *   Como su nombre lo indica asigna un valor (String) al atributo palabra.
        */
        public void asignaPalabra(String palabra){
             this.palabra = palabra;
        }


}
```

5.9 Candidatos Concatenación

La clase Candidatos Concatenación, es la encargada de asignarle a cada palabra un peso, un wav y como lo mencionábamos anteriormente, manda llamar a la clase fronteras, para saber en donde empieza y termina cada palabra.

Esta clase es llamada por constructor de oraciones, que será la encargada de crear las palabras o frases a sintetizar, para a su vez, mandarla a la clase principal, y generar el audio requerido.

```
public CandidatosConcatenacion(){

      wav="hello";
      contexto_central=new Fronteras();
      contexto_izquierdo=new Fronteras();
      contexto_derecho=new Fronteras();
      peso=0;
      num_palabra=0;
      codigo="000";

   }

   public CandidatosConcatenacion(String wav,Fronteras izq,Fronteras cen,Fronteras
der,int peso,int num_palabra,String codigo){
      asignaWavCandidato(wav);
      asignaContIzqCandidato(izq);
      asignaContDerCandidato(der);
      asignaContCenCandidato(cen);
      asignaPesoCandidato(peso);
      asignaNumPalabraCandidato(num_palabra);
      asignaCodigoCandidato(codigo);
   }

   //regresa el valor del wav candidato
   public String regresaWavCandidato(){
      return wav;
   }

   //regresa los valores del contexto izquierdo del wav candidato
   public Fronteras regresaContIzqCandidato(){
      return contexto_izquierdo;
   }
```

```
  //regresa los valores del contexto central del wav candidato
  public Fronteras regresaContCenCandidato(){
     return contexto_central;
  }

  //regresa los valores del contexto derecho del wav candidato
  public Fronteras regresaContDerCandidato(){
     return contexto_derecho;
  }

  //regresa el peso correspondiente al número de contextos que se encontraron en el
wav
  //candidato a síntesis
  public int regresaPesoCandidato(){
     return peso;
  }

  //regresa el num correspondiente a la palabra que es parte del wav candidato a síntesis
  public int regresaNumPalabraCandidato(){
     return num_palabra;
  }

  //regresa el código del wav candidato
  public String regresaCodigoCandidato(){
     return codigo;
  }

  //asigna el nombre del wav que es candidato a sintetizarse
  public void asignaWavCandidato(String wav){
     this.wav = wav;
  }

  //asigna el contexto izquierdo del wav que es candidato a sintetizarse
  public void asignaContIzqCandidato(Fronteras izq       ){
     contexto_izquierdo = izq;
  }

  //asigna el contexto derecho del wav que es candidato a sintetizarse
  public void asignaContDerCandidato(Fronteras der       ){
     contexto_derecho = der;
  }

  //asigna el contexto central del wav que es candidato a sintetizarse
  public void asignaContCenCandidato(Fronteras cen       ){
     contexto_central = cen;
  }

  //asigna el peso correspondiente a los contextos que se manejan en el wav que
  //es candidato a sintetizarse, puede manejar sólo 3 valores, 1 si tiene un sólo contexto
```

```
    //en este caso el central, 2 si maneja los contextos central-izquierdo o central-derecho,
    //3 si todos los contextos se encuentran presentes
    public void asignaPesoCandidato(int peso){
        this.peso = peso;
    }

    //asigna el numero de palabra que corresponde al contexto central que se esta evaluando,
    //este número proviene de la posición que ocupa la palabra que representa el contexto central
    //dentro de la frase a sintetizar. Por ejemplo en la trascripción fonética de las personas mueren,
    //osea pau -> las -> personas -> mueren -> pau. el # de palabra correspondiente a mueren es el 4.
    public void asignaNumPalabraCandidato(int num_palabra){
        this.num_palabra = num_palabra;
    }

    //el codigo es una simbología que sirve para identificar los contextos que se tienen presentes
    //para una palabra dada. Por ejemplo, "111" si tenemos los tres contextos, "010" si sólo se
    //tiene al contexto central, "110" si el contexto es central-izquierdo o "011" central-derecho
    public void asignaCodigoCandidato(String codigo){
        this.codigo = codigo;
    }
}
```

5.10 Interfaces del Sistema

Fig. 5.5 Frame de para importar corpus

Esta es la interface principal para empezar la incorporación del Corpus

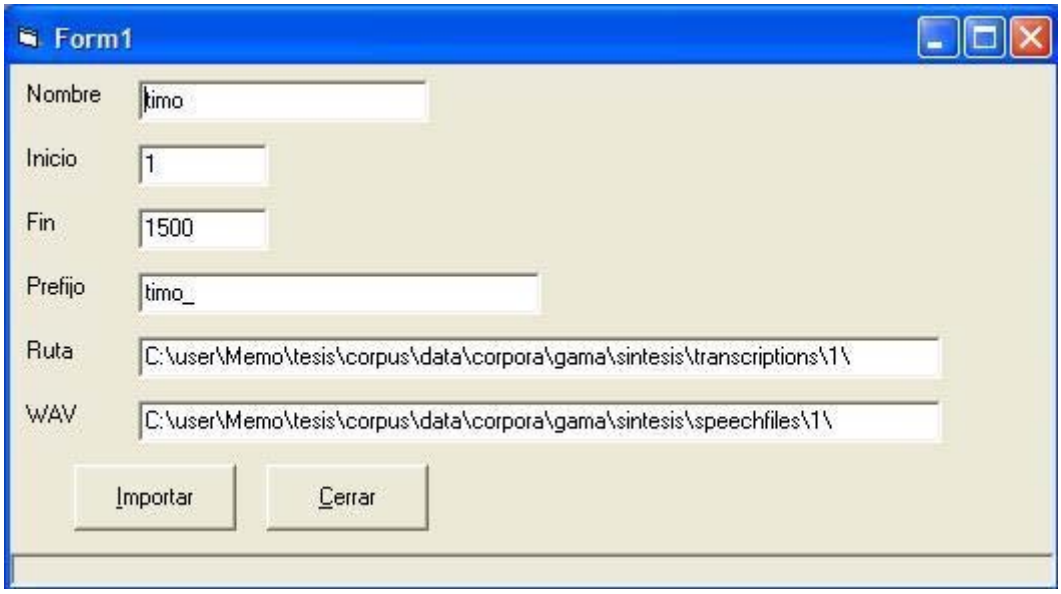

Fig. 5.6 Frame para importar los corpus a la base de datos.

Interfaz secundaria, aquí se proporcionan los datos del Corpus a importar.

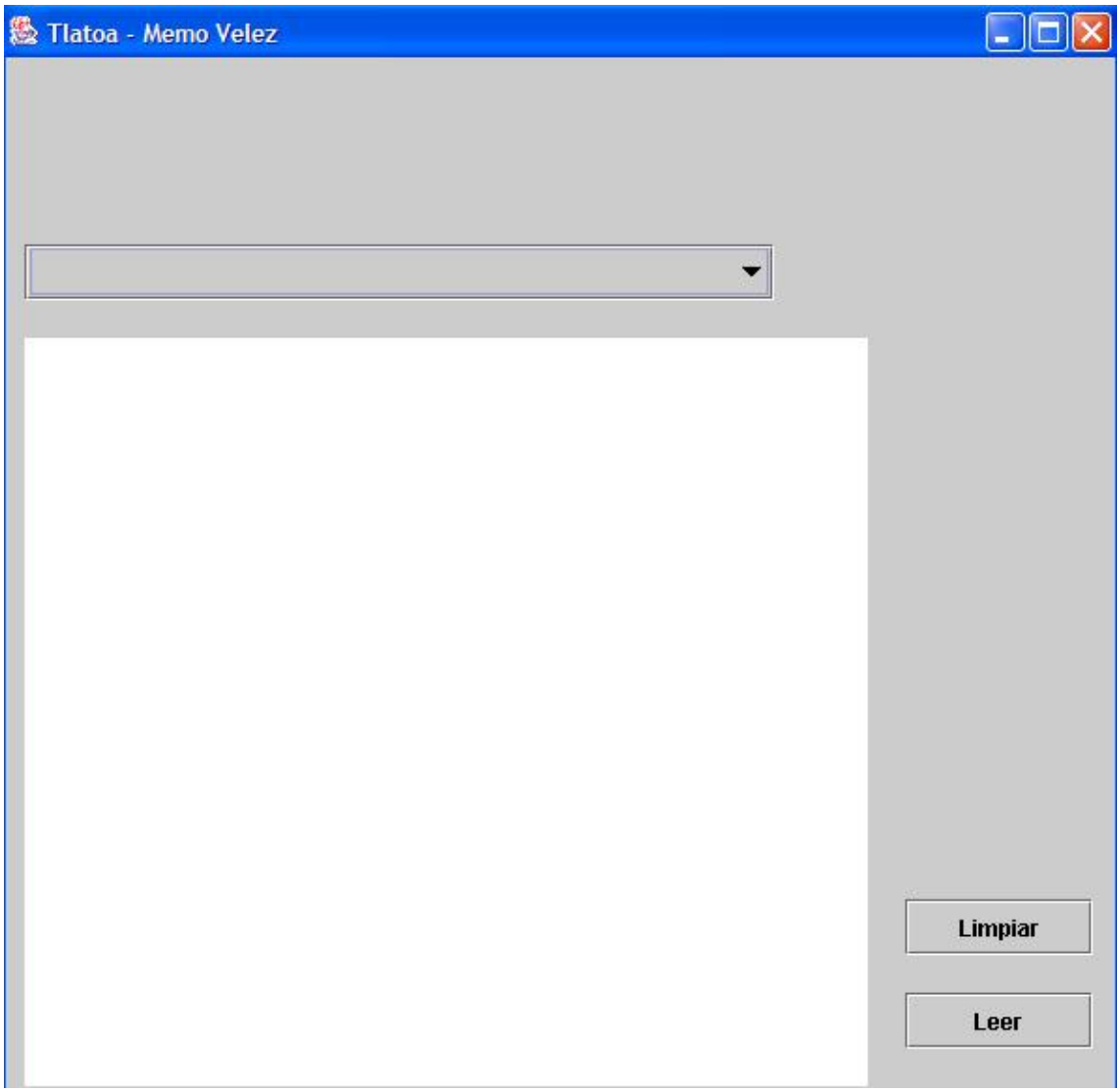

Fig. 5.7 Frame principal del Sintetizador

Esta es la Interfaz principal donde el usuario podrá escribir el texto que desee sintetizar.

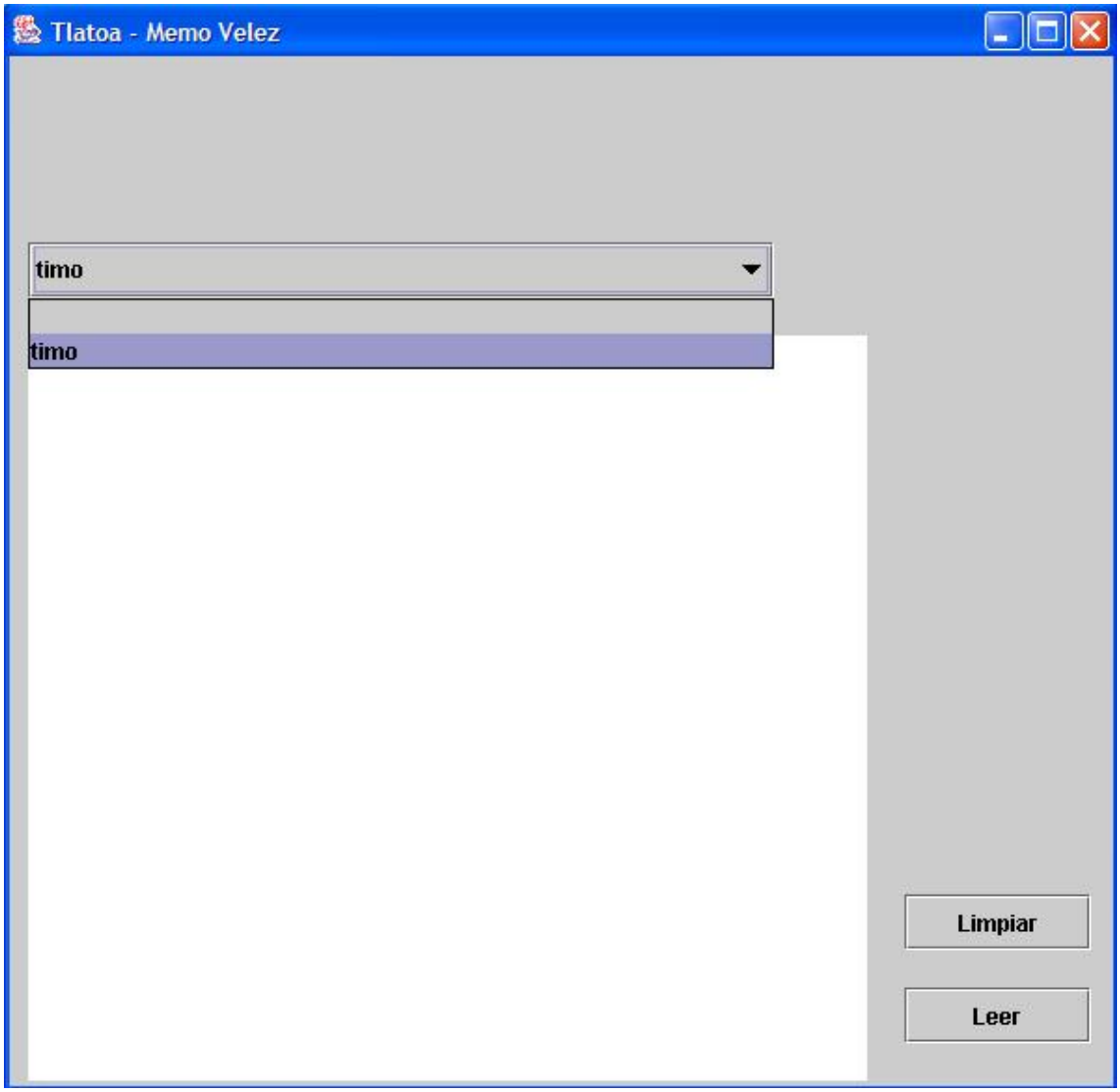

Fig. 5.8 Frame de Seleccion de Corpus

Esta es la interfaz donde el usuario selecciona un corpus para después introducir el texto que desee sintetizar, para después al oprimir el botón de leer, empezara todo el proceso de sintetizacion.

5.11 Resumen

En este capitulo se explica como se implemento el sistema, con las clases y métodos que llevan a cabo la labor de síntesis, desde la implementación de la base de datos, así como la importación del corpus a la misma, hasta la explicación de cada una de las clases y su implementación.