MANUAL TÉCNICO

La aplicación “Analizador Léxico” trata de un editor de texto que posea las funciones básicas para la edición de texto. La aplicación debe ser capaz de analizar el texto ingresado, representar la tabla de tokens respectiva de no existir ningún error, de existir el resultado deber ser una tabla de errores con sus respectivas líneas y columnas para identificar donde se encuentra dicho error. Además de lo anterior, la aplicación debe ser capaz de crear un diagrama de clases del texto ingresado, esto con la notación basada en UML. Por lo consiguiente se implementó un analizador léxico que fuera recorriendo carácter por carácter y agregándolo a dicha lista según su tipo: Palabras Reservadas, Identificadores, Signos especiales y delimitadores. Además, se hizo uso de “Graphviz” para la generación del diagrama de clases.

REQUERIMIENTOS MINIMOS

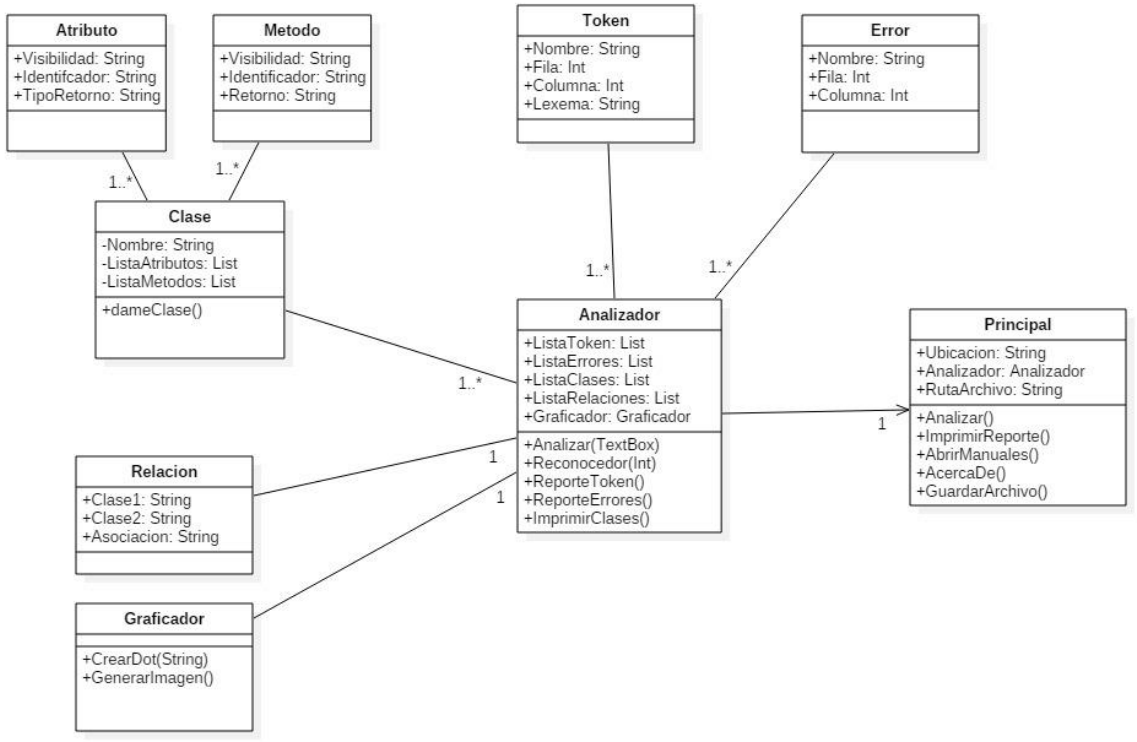
- Windows 7 o posterior
- .Net Framework 4.7.2
- Graphviz Instalado

PLATAFORMA DE DESARROLLO

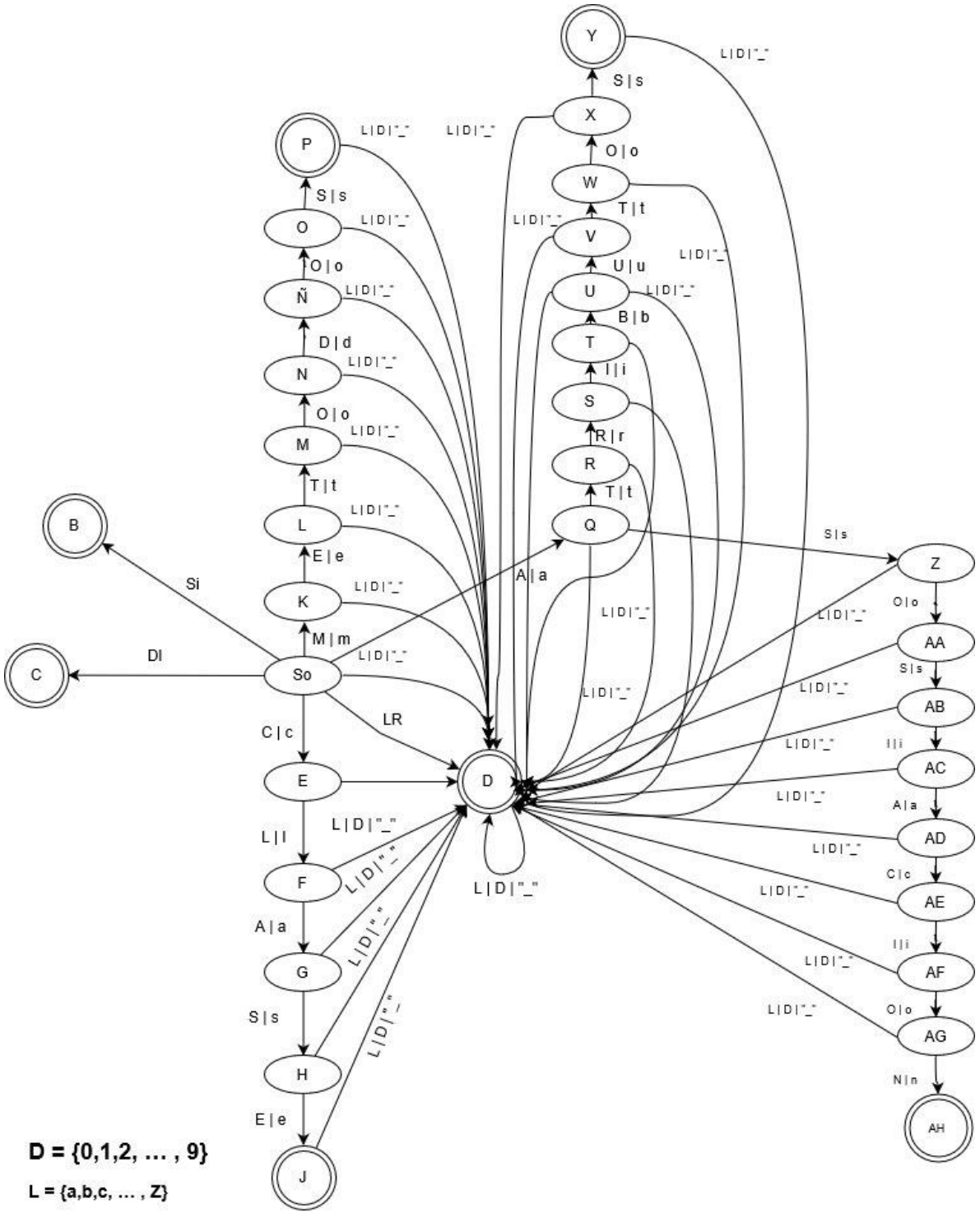
La aplicación se desarrolló bajo el lenguaje de programación Visual Basic, mediante el IDE Visual Studio Community 2017 Versión 15.8.1.

El Framework utilizado es Microsoft .Net Framework Versión 4.7.03056

DIAGRAMA DE CLASES



DFA



D = {0,1,2, … , 9}

L = {a,b,c, … , Z}

Lr = {L exepto A,a,C,c,M,m,N,n}

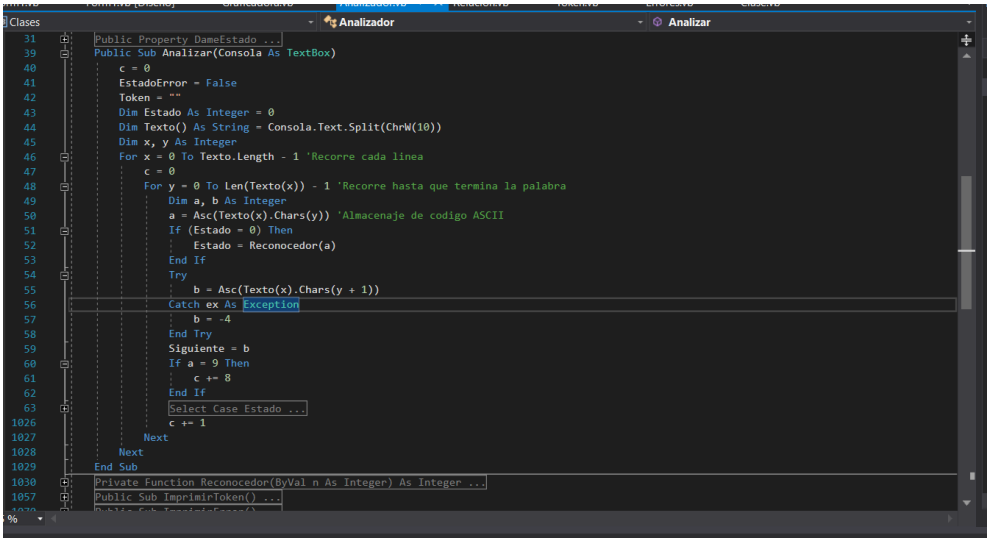
Si = { +,-,#,(,),[,] }

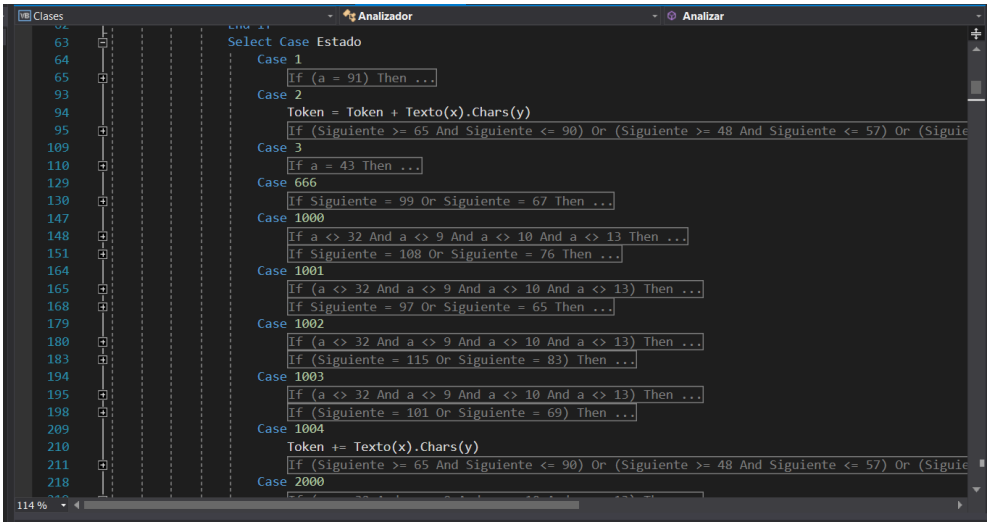
DI = { { , }, ; , : }

PUNTOS IMPORTANTES DEL CODIGO

Algunas de las partes más importantes del código son las siguientes:

1. La clase recibe por atributo un textbox, que es el que vamos a obtener del frame principal, en dicho textbox va el texto ingresado, posterior creamos un arreglo de String, agregando carácter por carácter. Recorremos la lista de línea a línea, carácter por carácter y vamos almacenando el carácter que viene y el carácter siguiente. Adicionalmente tenemos un método llamado “Reconocedor” el cual reconoce el tipo de carácter que viene y retorna un digito, este digito nos servirá para saber qué camino tomar en los Case siguientes, por facilidad se trabajó bajo el esquema del código ASCII. Cada case analiza lo que viene y sigue su camino, dependiendo si es palabra reservada, símbolo, identificador, etc. Cuando este llega a un estado de aceptación almacena el token, lo guarda en la lista de tokens y termina su recorrido. Cada case significa un estado del DFA anterior.





2. La clase Graficadora, es necesario tener instalado Graphviz, en esta ejecutamos lo que queremos diagramar, bajo el lenguaje de Graphviz y lo guardamos en un archivo *.dot este nos servirá para generar la imagen. A través de unos comandos de consola se genera la imagen, es para esto que sirve lo siguiente:

```
Public Sub GenerarImagen()
    Dim doPath As String = "D:\Program Files\Graphviz\bin\dot.exe"
    Dim cmd As String = doPath & " -Tjpg " & "UML.dot" & " -o " & "diagrama.jpg"
    Dim prog As VariantType
    prog = Shell(cmd, 1)
End Sub
```

Donde “doPath”, es la ruta donde está instalado Graphviz, y “cmd” es el comando a ejecutar en la consola, la parte de “ -Tjpg” y “-o” siempre viene, la cadena “UML.dot” es el nombre del archivo a graficar guardado previamente y “diagrama.jpg”, es el nombre del archivo imagen que vamos a generar.

REPOSITORIO DEL ARCHIVO

https://github.com/erflod5/Proyectos/tree/master/Analizador%20Lexico