Pseudo code to Java Translator: Instrument that Facilitates the encoding of this language

Traductor de Pseudocódigo a Java: Herramienta Didáctica que Facilita la Codificación de este Lenguaje

Genny Herazo y Miguel López
Universidad Simón Bolívar
Barranquilla - Atlántico

Keywords:
Translator, Pseudocode, Algorithm, Logic, Programming, Compiler

Abstract
Currently, developers, analysts and software designers, have tools that allow them to devise solutions powerful, modern and affordable to users, thus achieving the effectiveness of its activities. As we know, the basis for application development are the source codes, which in turn are designed by software algorithms. It is here where most beginners (application development), are disadvantages when coding their algorithms in programming languages such as Java, PHP, C++. Among others. Given the weaknesses when developing applications, there is a way to alleviate the need for anyone starting their planning stage in the Java language can be resolved simply and teaching their encodings. The solution is basically a pseudo analyzer, which then carry out a lexical and syntactic analysis, which translates to the judgments used in the Java language. Thus, those with little knowledge of programming on this platform may learn more easily on sentences in this language

Palabras clave:
Traductor, Pseudocódigo, Algoritmo, Lógica, Programación, Compilador

Resumen
En la actualidad, los desarrolladores, analistas y diseñadores de software, cuentan con herramientas que les permiten idear soluciones informáticas potentes, modernas y asequibles a los usuarios; logrando así la eficacia de sus actividades. Como sabemos, la base para el desarrollo de aplicaciones son los códigos fuentes, que a su vez están ideados a través de algoritmos lógicos. Es en este punto, donde la mayoría de los principiantes (en desarrollo de aplicaciones), encuentran inconvenientes, al momento de codificar sus algoritmos en lenguajes de programación, tales como Java, PHP, C++. Entre otros. Teniendo en cuenta las falencias a la hora de desarrollar aplicaciones, se plantea una forma de aliviar la necesidad de todo aquel que iniciando su etapa de programación en el lenguaje Java pueda resolver de manera sencilla y didáctica sus codificaciones. La solución básicamente es un analizador de pseudocódigo, que luego de realizar un análisis léxico y sintáctico, lo traduzca a las sentencias utilizadas en el lenguaje Java. De esta forma, aquellos que tienen pocos conocimientos en esta plataforma de programación, podrán aprender de forma más sencilla las sentencias utilizadas en este lenguaje.

I. INTRODUCCIÓN

La programación es un campo esencial para los estudiantes de informática y por ende es necesario dedicar un espacio de estudio a estas áreas de programación que son claves durante los primeros años de la formación profesional.

Se ha observado que el estudiante al iniciar una respectiva programación en un lenguaje determinado, se le complica el proceso de la codificación. Debido a esto se analizó el hecho en la Universidad Simón Bolívar y se tomó como referencia el lenguaje Java; donde se obtuvo como resultado dificultad para programar en esta plataforma ya que no cuentan con un medio o una herramienta alterna de interacción didáctica donde el estudiante pueda desenvolverse aún más con este lenguaje y que complementen lo enseñado en clases el cual puedan mejorar su capacidad analítica y creadora a la hora de codificar en Java. Teniendo en cuenta lo anterior encontramos que algunas organizaciones han desarrollado herramientas que permiten traducir los códigos de un lenguaje de programación a otro, y de esta forma familiarizarse con un lenguaje de programación partiendo de otro, pero son demasiado escasas y poco conocidas, las herramientas que permiten traducir pseudocódigo.

Para mitigar esta situación de acuerdo a lo analizado se ha trabajado en el desarrollo de un traductor de pseudocódigo a lenguaje Java como herramienta didáctica para facilitar la codificación de este lenguaje. Aportando así un medio de solución a los estudiantes que están dando sus primeros pasos o deseen familiarizarse aun más con la programación en Java; permitiendo de esta manera que adquieran habilidades y destrezas para codificar.

Este artículo contiene cada uno de los componentes esenciales que hacen parte del proceso de una traducción; el cual está desarrollado basándose en conceptos teóricos como: lenguajes de programación, traductores de programa, mostrando así la estructura que forma parte del traductor destacando sus etapas, la técnica utilizada para su construcción, la metodología que muestra el procesamiento de la herramienta, el medio que se usa para hacer la respectiva traducción, y finalmente el lenguaje de programación en que traduce la herramienta.

II. LENGUAJES DE PROGRAMACIÓN

Un lenguaje de programación es un lenguaje que puede ser utilizado para controlar el comportamiento de una máquina, particularmente una computadora. [1]. Una computadora funciona bajo control de un programa el cual debe estar almacenado en la unidad de memoria; tales como el disco duro.

Los lenguajes de programación facilitan la tarea de programación, ya que disponen de formas adecuadas que permiten ser leídas y escritas por personas, a su vez resultan independientes del modelo de computador a utilizar. Algunos ejemplos de lenguajes de programación son: Visual Basic, Java, C++, PHP, Matlab, entre otros. [2]. Cabe destacar que los lenguajes de programación contienen un conjunto de reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos, respectivamente. [1]

Existen diferentes clases o tipos de lenguajes de programación:

- Lenguaje máquina

Es el lenguaje de programación que entiende directamente la computadora o máquina. Este lenguaje de programación utiliza el alfabeto binario, es decir, el 0 y el 1. Con estos dos únicos dígitos, conocidos como bits, forma lo que se conoce como cadenas binarias (combinaciones de ceros y unos) son con las que se escriben las instrucciones que el microprocesador de la computadora entiende nuestra peticiones. Fue el primer lenguaje de programación. Este tipo de lenguaje de programación dejo de utilizarse por su gran dificultad y por la facilidad para cometer errores. [3]

- Lenguaje de bajo nivel (ensamblador)

Son mucho más fáciles de utilizar que el lenguaje máquina, pero dependen mucho de la máquina o computadora como sucedía con el lenguaje máquina. El lenguaje ensamblador fue el primer lenguaje de programación que trato de sustituir el lenguaje máquina por otro mucho más parecido al de los seres

humanos. En este lenguaje se conoce como programa fuente a un programa de instrucciones escrito en lenguaje ensamblador por el programador, y programa objeto es la traducción a lenguaje máquina del programa fuente.

Los lenguajes de este tipo pueden crear programas muy rápidos, pero son difíciles de aprender, son específicos de cada procesador, si nos llevamos el programa a otro computador será preciso reescribir el programa desde el comienzo. [3]

- Lenguajes de alto nivel

Este tipo de lenguajes de programación son independientes de la máquina, lo podemos usar en cualquier computador con muy pocas modificaciones o sin ellas, son muy similares al lenguaje humano, pero precisan de un programa interprete o compilador que traduzca este lenguaje de programación de alto nivel a uno de bajo nivel como el lenguaje de máquina que la computadora pueda entender.

Los lenguajes de programación de alto nivel son más fáciles de aprender porque se usan palabras o comandos del lenguaje natural, como por ejemplo del inglés [3].

III. TRADUCTORES DE PROGRAMAS

Los traductores son programas que traducen los programas en código fuente, escritos en lenguajes de alto nivel, a programas escritos en lenguaje máquina. [5]

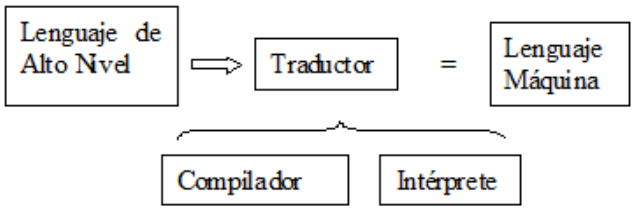

Figura 1. Esquema del proceso de un traductor

Existen distintos tipos de traductores, entre ellos destacan:

- Ensambladores

Es un tipo de traductor que convierte programas escritos en lenguaje ensamblador en programas escritos en código máquina. [4]

- Intérpretes

Los intérpretes no producen un lenguaje objetivo como en los compiladores. Un intérprete lee el código como está escrito e inmediatamente lo convierte en acciones; es decir, lo ejecuta en ese instante.

Existen lenguajes que utilizan un intérprete (como por ejemplo JAVA) que traduce en el instante mismo de lectura el código en lenguaje máquina para que pueda ser ejecutado. La siguiente Figura 2 muestra el funcionamiento de un intérprete. [5]

Figura 2. Funcionamiento de un intérprete

- Compiladores

Un compilador es un programa que lee el código escrito en un lenguaje (lenguaje origen), y lo traduce en un programa equivalente escrito en otro lenguaje (lenguaje objetivo). Como una parte fundamental de este proceso de traducción, el compilador le hace notar al usuario la presencia de errores en el código fuente del programa. Ver Figura 3. [5]

Figura 3. Funcionamiento de un compilador

Los lenguajes C y C++ son lenguajes que utiliza un compilador. El trabajo del compilador y su función es llevar el código fuente escrito en C/C++ a un programa escrito en lenguaje máquina. Entrando en más detalle, un programa en código fuente es compilado obteniendo un archivo parcial (un objeto) que tiene extensión obj. Luego el compilador invoca al "linker" que convierte al archivo objeto en un ejecutable con extensión exe; este último archivo es un archivo en formato binario (ceros y unos) y puede funcionar por sí sólo. Además, el compilador al realizar su tarea realiza también una comprobación de errores en el programa; es decir, revisa que todo esté en orden. Por ejemplo, variables y funciones bien definidas, todo lo referente a cuestiones sintácticas, etc. Está fuera del alcance del compilador que, por ejemplo, el algoritmo utilizado en el problema funcione bien. La siguiente Figura 4 muestra los pasos para tener un programa ejecutable desde el código fuente. [5]

Figura 4. Pasos del compilador

Los compiladores difieren de los intérpretes en varios aspectos:

Un programa que ha sido compilado puede correr por sí sólo, pues en el proceso de compilación se lo transformo en otro lenguaje (lenguaje máquina). Un intérprete traduce el programa cuando lo lee, convirtiendo el código del programa directamente en acciones. La ventaja del intérprete es que dado cualquier programa se puede interpretar en cualquier plataforma (sistema operativo). En cambio, el archivo generado por el compilador solo funciona en la plataforma en donde se le ha creado. Sin embargo, hablando de la velocidad de ejecución, un archivo compilado es de 10 a 20 veces más rápido que un archivo interpretado. [5]

IV. ETAPAS EN LA CONSTRUCCIÓN DEL TRADUCTOR

Para construir el traductor se tuvo en cuenta las siguientes etapas:

A. Análisis léxico

Es el encargado de leer caracter por caracter de un archivo (que contenga código fuente escrito en algún lenguaje de programación específico) y construir elementos léxicos llamados patrones que serán utilizados posteriormente por un analizador sintáctico. Un patrón es una pareja ordenada compuesta por un token y un lexema. Un token es el elemento léxico del lenguaje, es decir el símbolo terminal de una gramática libre de contexto (GLC). Y por último, un lexema es la secuencia de caracteres que coinciden con un token. [6]

El siguiente, es un ejemplo de análisis léxico para el lenguaje Java:

```
public class HolaMundo {
 public static void main(String[] args){
    System.out.println("Hola mundo");
 }
}
```

Tras procesar el código con un analizador léxico, este realiza lo siguiente:

```
PalabraReservada, public
PalabraReservada, class
Texto, HolaMundo
LlaveIzq,{
PalabraReservada, public
PalabraReservada, static
PalabraReservada, void
PalabraReservada, main
ParentesisIzq, (
```

```
PalabraReservada, String
CorcheteIzq, [
CorcheteDer,]
PalabraReservada, args
ParentesisDer, )
LlaveIzq, {
PalabraReservada, system.
PalabraReservada, out.
PalabraReservada, println
ParentesisIzq,(
Texto, "Hola mundo"
ParentesisDer,)
PuntoYcoma, ;
LlaveDer, }
LlaveDer, }
```

B. Análisis sintáctico

Comprueba que las sentencias que componen el texto fuente sean correctas en el lenguaje correspondiente, creando una representación interna que corresponde a la sentencia analizada. De esta manera se garantiza que sólo serán procesadas las sentencias que pertenezcan al lenguaje fuente. Así como en las demás etapas, durante el análisis sintáctico, se van mostrando los errores que se encuentran. [7]

V. TÉCNICA UTILIZADA EN LA TRADUCCIÓN

La técnica que se utilizó para la verificación de la secuencia de las cadenas en la traducción es la gramática independiente del contexto (GIC); ya que ésta describe a la mayoría de los lenguajes de programación, de hecho, la sintaxis de la mayoría de lenguajes de programación está definida mediante gramáticas libres de contexto. Por otro lado, estas gramáticas son suficientemente simples como para permitir el diseño de eficientes algoritmos de análisis sintáctico que, para una cadena de caracteres dada determinen como puede ser generada desde la gramática.

- Gramática independiente del contexto (GIC)

Una Gramática independientes del contexto (GIC) es una gramática formal en la que cada regla de producción es de la forma: Exp → x donde Exp es un símbolo no terminal y x es una cadena de terminales y/o no terminales. El término independiente del contexto se refiere al hecho de que el no terminal Exp puede siempre ser sustituido por x sin tener en cuenta el contexto en el que ocurra. Un lenguaje formal es independiente de contexto si hay una gramática libre de contexto que lo genera, este tipo de gramática fue creada por Backus-Naur y se utiliza para describir la mayoría de los lenguajes de programación. [8]

Una GIC está compuesta por 4 elementos:

1. Símbolos terminales (elementos que no generan nada). [8]

2. No terminales (elementos del lado izquierdo de una producción, antes de la flecha "->"). [8]
3. Producciones (sentencias que se escriben en la gramática). [8]

4. Símbolo inicial (primer elemento de la gramática). [8]

Ahora, es importante decir que la gramática LL es descendente de la gramática independiente del contexto (GIC); a través de la gramática LL obtenemos un análisis sintácticamente correcto aplicando cada una de las reglas que esta propone. A continuación se explica detalladamente la gramática LL:

- Gramática LL

Se caracteriza porque tiene la función de predecir qué producción debe escoger para la evaluación de las cadenas. Se debe saber y tener en cuenta que no todas las gramáticas se pueden trabajar con el LL.

A. Teorema

Una gramática es LL si y solo si:

Dadas 2 producciones de la forma $A \longrightarrow \infty$, $A \longrightarrow \beta$ se debe cumplir:

a) Prim(∞) ⋂ Prim (β)={ ϵ }
b) Si ∞ ⟶ ϵ y β⟶

Si ∞⟶ ϵ y β ⟶ ϵ, entonces Prim (∞) ⋂ Sigte(A) = { ϵ }

Reglas de Primero: (Prim)

a) Si X ⟶ ϵ, entonces (ϵ) está en Prim(X)

b) Si X ⟶ a, entonces (a) está en Prim(X)

c) Si X ⟶ Y1Y2…Yn, entonces Prim (Y1) está en Prim(X)

B. Una Gramática no es LL por las siguientes razones

- *Gramática es ambigua*
- Recursividad por izquierda
- Factorizable por izquierda

1. Si la gramática es ambigua se debe quitar la ambigüedad y para quitarla se debe reescribir la gramática.

2. Debemos eliminar la recursividad por izquierda.

3. Se debe factorizar por izquierda.

C. Tabla LL

TABLA LL						
T / NT	+	*	dig	(	)	$
E			E → TE′	E → TE′		
E′	E′→+TE′				E′ → ϵ	E′ → ϵ
T			T →FT′	T →FT′		
T′	T′ → ϵ	T′→*FT′			T′ → ϵ	T′ → ϵ
F			F → dig	F → (E)		

Figura 5. Tabla para la gramática LL

A continuación se muestra la Figura 6 donde se observa la pila y las producciones formadas en ella.

PILA	CADENA	PRODUCCION
$E	dig*dig+dig $	
$E′T	dig*dig+dig $	E→TE′ ←
$E′T′F	dig*dig+dig $	T→FT′
$E′T′dig	dig*dig+dig $	F→dig
$E′T′	*dig+dig $	
$E′T′F*	*dig+dig $	T′→*FT′
$E′T′F	dig+dig $	
$E′T′dig	dig+dig $	F→dig
$E′T′	+dig $	
$E′	+dig $	T′→ ϵ
$E′T+	+dig $	E′→+TE′
$E′T	dig $	
$E′T′F	dig $	T→FT′
$E′T′dig	dig $	F→dig
$E′T′	$	
$E′	$	T′→ ϵ
$	$	E′→ ϵ

Figura 6. Pila

VI. MEDIO PARA REALIZAR LA TRADUCCIÓN EN LA HERRAMIENTA

La herramienta realiza la traducción por un medio de expresión algorítmica denominado pseudocódigo ya que este por su uso cómodo y fácil nos lleva a interactuar con la herramienta en nuestro propio lenguaje natural.

Pseudocódigo

Es un lenguaje de especificación de algoritmos. El uso de tal lenguaje hace que el paso de codificación final es decir, a un lenguaje de programación sea relativamente fácil Un pseudocódigo es una especie de borrador del algoritmo, ya que no es directamente ejecutable sino que debe ser traducido a un lenguaje de programación.

La ventaja de usar pseudocódigos reside en el hecho de que permite elaborar la lógica del algoritmo independientemente de las instrucciones del lenguaje. Además, el pseudocódigo es fácilmente traducible a lenguajes de programación de alto nivel ya que para implementar un pseudocódigo se usan palabras similares a las usadas en los lenguajes de programación (Inicio, Fin, Para, Si-entonces-sino, mientras -hasta).

La escritura de un pseudocódigo exige normalmente el uso de sangrías que permiten visualizar la estructura del mismo. Debe comenzar con la palabra inicio y terminar con la palabra fin. Sólo se escribe una instrucción o acción por línea. [9]

VII. METODOLOGÍA DEL PROCESO DE LA TRADUCCIÓN

La herramienta realiza el proceso de la traducción de la siguiente manera:

1. El usuario define el planteamiento del ejercicio (enunciado del problema) a solucionar.

2. Se escribe en pseudocódigo el ejercicio o problema en el modulo especificado. No se debe olvidar tener

en cuenta las reglas específicas para escribir en pseudocódigo.

3. Compila el respectivo pseudocódigo usando una estructura basada en la gramática LL; es decir evalúa las cadenas de izquierda a derecha para que sean validadas cada una de las líneas de instrucción escritas. Es en esta parte donde se verifica si la escritura hecha en el pseudocódigo es correcta o incorrecta.

4. Después de que las líneas escritas han sido verificadas entonces se procede a ejecutar la herramienta de traducción en el lenguaje previamente especificado y la herramienta internamente procede a buscar en la base de datos y asocia el pseudocódigo escrito a una sentencia equivalente en Java.

5. Como resultado final se obtiene la traducción del pseudocódigo respectivo.

VIII. LENGUAJE ALGORITMICO

Es una serie de símbolos y reglas que se utilizan para describir de manera explícita un proceso, que servirán de apoyo para describir las soluciones que aquí se plantean. [10]

Teniendo en cuenta la forma en que describen el proceso, existen dos tipos de lenguajes algorítmicos:

- Gráficos: Es la representación gráfica de las operaciones que realiza un algoritmo (diagrama de flujo). [10]
- No Gráficos: Representa en forma descriptiva las operaciones que debe realizar un algoritmo (pseudocódigo). [10]

A. Definición de algoritmo

La palabra algoritmo se deriva de la traducción al latín de la palabra árabe Alkhowarizmi, nombre de un matemático y astrónomo árabe que escribió un tratado sobre la manipulación de números y ecuaciones en el siglo IX.

Se define como una serie de pasos organizados que describen el proceso que se debe seguir, para dar solución a un problema específico. [10]

Las principales características que debe tener un buen algoritmo son:

- Debe tener un punto particular de inicio. [10]
- Debe ser completamente definido y no debe permitir dobles interpretaciones. [10]
- Debe ser general, es decir, soportar la mayoría de las variantes que se puedan presentar en la definición del problema. [10]
- Debe ser finito en tamaño y tiempo de ejecución. [10]
- Debe ser legible, claro y fácil de interpretar y entender. [10]

IX. LENGUAJE DE PROGRAMACIÓN EN QUE TRADUCE LA HERRAMIENTA

La herramienta traduce el pseudocódigo en el siguiente lenguaje orientado a objetos:

Lenguaje Java

Java es toda una tecnología orientada al desarrollo de software con el cual podemos realizar cualquier tipo de programa. Hoy en día, la tecnología Java ha cobrado mucha importancia en el ámbito de Internet gracias a su plataforma J2EE. Pero Java no se queda ahí, ya que en la industria para dispositivos móviles también hay una gran acogida para este lenguaje.

La tecnología Java está compuesta básicamente por dos elementos: el lenguaje Java y su plataforma. Con plataforma nos referimos a la máquina virtual de Java (Java Virtual Machine).

Una de las principales características que favoreció el crecimiento y difusión del lenguaje Java es su

capacidad de que el código funcione sobre cualquier plataforma de software y hardware. Esto significa que nuestro mismo programa escrito para Linux puede ser ejecutado en Windows sin ningún problema. Además es un lenguaje orientado a objetos que resuelve los problemas en la complejidad de los sistemas, entre otras. [11]

El siguiente es un ejemplo práctico hecho en lenguaje Java:

Ejemplo 1 práctico en Java: Suma de dos números

```
Pseudocódigo: Suma de dos números
int num1,num2,sum;
inicio
escriba "Digite el primer número";
lea num1;
escriba "Digite el segundo número";
lea num2;
sum=num1+num2;
escriba "La suma de los números es"sum;
fin
```

```
Lenguaje Java: Suma de dos números
public class SumarNumeros {
 public static void main(String args[]){
 String Numero1;
 String Numero2;
 int Numero1;
 int Numero2;
 int suma;
 Numero1 = JOptionPane.showInputDialog(
  "Ingrese el entero 1" );
 Numero2 = JOptionPane.showInputDialog(
  "Ingrese el entero 2" );
 Numero1 = Integer.parseInt( Numero1 );
 Numero2 = Integer.parseInt(Numero2 );

 suma = Numero1 + Numero2;

 JOptionPane.showMessageDialog (null,
  "La suma es " + suma, "Resultado,",
 JOptionPane.PLAIN_MESSAGE );
     System.exit( 0 );
  }
}
```

X. LENGUAJE UTILIZADO PARA LA CONSTRUCCIÓN DE LA HERRAMIENTA

El lenguaje de programación utilizado para construir el traductor fue Visual Basic. NET ya que este lenguaje posee características y funcionalidades aptas para la elaboración y diseño de aplicaciones.

- Lenguaje Visual Basic.NET

Visual Basic.NET es en un lenguaje de programación orientado a objetos; podemos considerar que este lenguaje es una evolución de Visual Basic implementada sobre el Framework .NET.

La plataforma .NET es un componente de los sistemas operativos Windows, que permite el desarrollo, la liberación y la ejecución de aplicaciones. La plataforma posee un conjunto de herramientas de desarrollo y lenguajes de programación (de propósito general, orientados a objetos, de tercera generación, de alto nivel y compilación a código intermedio), que nos permiten utilizar todos los recursos disponibles en la computadora a través de una librería de clases común, con la que se pueden desarrollar aplicaciones de Consola, basadas en Windows, y para la web, que utilizan protocolos abiertos para la interacción entre los elementos que las componen.[12]

XI. RESULTADOS

A continuación se muestran los resultados obtenidos del traductor de pseudocódigo construido en Visual Basic.NET

En la siguiente figura 7 se observan 3 recuadros los cuales representan lo siguiente:

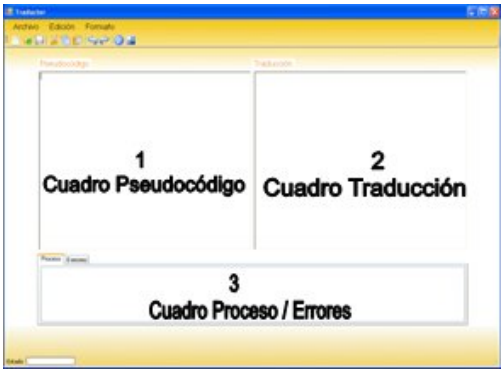

Figura 7. Interfaz Traductor

1. Cuadro del Pseudocódigo: en esta parte escribimos el algoritmo en pseudocódigo.

2. Cuadro de la traducción: aquí se muestra la traducción del respectivo pseudocódigo.

3. Cuadro de proceso/Errores: aquí se muestra los errores que se puedan producir en el momento de compilar el algoritmo.

En la figura 8 se muestra en el tercer cuadro la barra de errores allí despliega la descripción del error, la línea en que está el error y la columna.

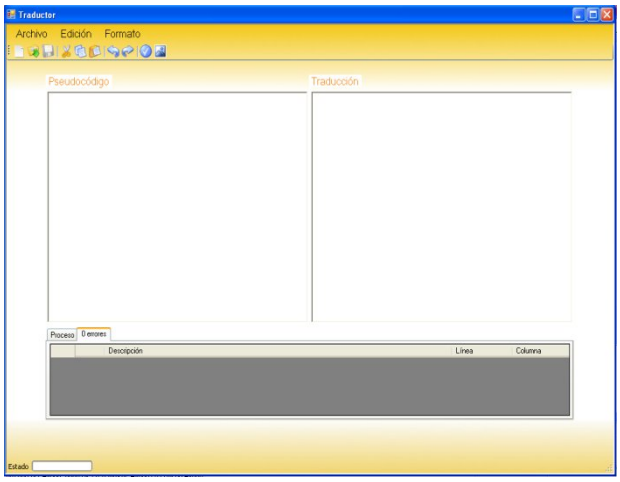

Figura 8. Barra de errores

XII. CONCLUSIÓN

La programación es un campo que hace parte durante los primeros años de un estudiante de sistemas y debido a esto es muy útil poder contar con una herramienta didáctica de aprendizaje como lo es el traductor de pseudocódigo a lenguaje de programación Java ya que ésta facilitaría la codificación en este lenguaje además resolvería los problemas algorítmicos de una manera sencilla y amigable; permitiendo obtener de esta manera resultados satisfactorio para aquel que desee hacer uso práctico de la herramienta.

La herramienta destaca su uso importante en el campo de programación y está diseñada para estudiantes universitarios, tecnólogos, técnicos que se encuentren estudiando carreras afines de sistemas y viendo asignaturas tales como: programación, informática, lógica y diseño de programación, entre otras donde sea posible aplicar el uso del traductor.

Cabe mencionar que esta herramienta aún no se le ha practicado prueba piloto por ello no se tiene experiencia alguna de su uso en el entorno educativo. Teniendo en cuenta esto es necesario que se le apliquen a esta herramienta pruebas para así obtener como resultados verídicos el buen funcionamiento de ésta. Por eso quedará soporte de lo efectuado en esta herramienta para que se practiquen futuramente estas pruebas.

REFERENCIAS

[1]El mundo informático. Fecha Publicación: 05-05-2005. Disponible en: http://jorgesaavedra.wordpress.com/2007/05/05/lenguajes-de-programacion/

[2]Lenguajes de programación. Año Publicación: 2009. Disponible en: http://www.lenguajes-de programacion.com/lenguajes-de-programacion.shtml

[3]Articulandia. Clasificación de los lenguajes de programación. Disponible en: http://www.articulandia.com/premium/article.php/06-09-2006Clasificacion-de-los-lenguajes-de-programacion.htm

[4]Teleformación. Tutorial Introductorio. Disponible en: http://teleformacion.edu.aytolacoruna.es/PASCAL/document/intro.htm

[5]Instituto Técnico de Celaya. Traductores y compiladores. Disponible en: www.iqcelaya.itc.mx/~vicente/Programacion/TradComp.doc -

[6]Analizador léxico. Fecha Publicación: 01-07-2004. Disponible: http://valar.wordpress.com/2004/07/01/analizador-lexico/

Informe técnico:

[7]C.A. Vanegas, "Compilador y traductor de pseudocódigo para la lógica de programación (CompiProgramación)", UNAM,México. Fecha Publicación: Primer semestre de 2005. Disponible en: http://profesores.fib.unam.mx/tanya/CPI/Documentos/conciencias6.pdf

[8]Connexions. Gramática independiente del contexto, ejemplos y ejercicios. Disponible en: http://cnx.org/content/m16320/latest/

[9]Escuela Técnica Nº 17 Cornelio Saavedra. Generalidades sobre los pseudocódigos. Disponible en: http://www.adrianpelliza.com.ar/pdf2008/Computadoras/Pseudoc%C3%B3digos.pdf

[10]Dirección nacional de servicios académicos virtuales. Cursos Sede Manizales. Análisis y diseños de algoritmos. Disponible en: http://www.virtual.unal.edu.co/cursos/sedes/manizales/4060024/Lecciones/Capitulo%20I/conceptos.htm

[11]Ciberaula Java. Artículo Java. Año publicación: 2006. Disponible en: http://java.ciberaula.com/articulo/que_es_java/

Libro:

[12]J. F. Ramírez y F. Salazar "Aprenda Practicando Visual Basic 2005 Usando Visual Studio" Cap. 1, p 5. Pearson Educación México S.A. Edit. División Computación. Disponible en línea: http://books.google.com.co/books?id=Nx1aQ_qbTNoC&pg=PA54&dq=visual-basic.net+espa%C3%B1ol&lr=#v=onepage&q=visual-basic.net%20espa%C3%B1ol&f=false