Instituto Tecnológico de Costa Rica

Escuela de Ingeniería en Computación
Sede Alajuela

Compiladores e Intérpretes

Profesor:
Esteban Arias Mendez

Lenguaje MicroPerl

Esteban Segura Benavides 201156221
Marvin Carranza Romero 201131292

Mayo 2015

Abstract:
In this article we present the main results of the final project of Compiladores e Interpretes class, in the article we describe the MicroPerl language, the functionality of the language, main characteristics, technical information andr tools used for build the main parts of the language, also comments and future work related to improve the MicroPerl language .

Tabla de Contenidos

1 Descripción del lenguaje

El lenguaje elegido por el equipo de trabajo es un subconjunto del lenguaje Perl. Perl es un lenguaje de programación interpretado de propósito general creado por Larry Wall en 1987, desarrollado originalmente para la manipulación de texto, y que ahora se utiliza para una amplia gama de tareas. Perl toma las mejores características de lenguajes como C, awk, sed, sh y BASIC entre otros, soporta tanto programación procedimental como orientada a objetos. Un programa en Perl consiste en una secuencia de declaraciones y sentencias, permite ciclos, subrutinas y otras estructuras de control para saltar dentro del código. (Perl Tutorial, s.f.)
El objetivo es desarrollar un lenguaje con las caracteristias reducidas de Perl para el curso de Compiladores e Interpretes, el lenguaje llamado MicroPerl es un lenguaje interpretado y es capaz de ejecutar un conjunto de instrucciones tales como ciclos, condiciones, operaciones aritmeticas,operacones logicas y definicion de funciones, asi como manejar diferentes tipos de variables, tambien incluye una serie de funciones propias del lenguaje a dispocion del usuario, para que este las utiliza cuando las considere necesario.

2 Características principales del lenguaje

En comparación con otros lenguajes Perl resulta ser un lenguaje con una sintaxis algo complicada por los símbolos e instrucciones que tienden a ser un poco entendible en algunos casos, sin embargo, después de cierto tiempo de utilizar el lenguaje, resulta intuitivo y muy cómodo para realizar las diferentes tareas que se necesitan de acuerdo a las funcionalidades que ofrezca Perl.Una caracteristica de Perl es la necesidad de utilizar un MY para declarar las variables, esto a pesar de ser opcional, en Perl existe una librería que hace obligatorio la colocación de la palabra reservada MY antes de declarar una variable, esa característica, se conserva en el lenguaje MicroPerl. Otra característica propia de Perl es la necesidad de anteponer un símbolo delante del identificador de la variable, para indicarle de forma explicita el tipo de variable. El lenguaje Perl maneja 3 tipos de símbolos para indicar el tipo de la variable el $ (dolar) el cual indica que la variable puede ser de tipo Entero o una Cadena de caracteres, el @ delante del identificador de la variable para indicar que esa variable sera un arreglo de elementos (Lista de cadenas o enteros) y por ultimo el símbolo % delante de un identificador de la variable indicando que la misma sera de tipo Hash (llave-valor). En el lenguaje MicroPerl se tomaron 2 de esos tres tipos de variables para ser utilizados en el lenguaje, los cuales son el $ y el @ respectivamente. Se decidio realizar un subconjunto de Perl, con miras de volver la sintaxis del lenguaje a una sintaxis mas amigable, pero manteniendo las funcionalidades que el lenguaje ofrece. El manejo de arreglo resulta muy sencillo con Perl , así como las diferentes funcionales que ofrece para manejar listas, funcionalidades que se mantienen en MicroPerl pero con una sintaxis mas sencilla, intentado que resulte mas intuitivo para el usuario.

3 Información técnica del lenguaje

Como se mencionó anteriormente, el Lenguaje MicroPerl es un subconjunto de Perl, y este a su vez tiene una sintaxis muy similar a C, sintaxis que se mantiene en MicroPerl, resultando muy similar las instrucciones de ciclos (for, while, foreach, etc.), así como las instrucciones de decisión (if, elsif, switch), manteniendo las mismas palabras reservadas para realizar los ciclos y toma de decisiones necesarias. También se mantienen otras palabras reservada como el break. Por otro lado MicroPerl toma cierto tipo de instrucciones cambiando un poco las sintaxis de la misma tal es el caso del print, función utilizada para mostrar un mensaje en consola, esta instrucción se busco mantener lo mas simple posible ya que se consideró que resulta de uso frecuente en los programas, otra funciones de manejo de cadenas o listas, tambien mantienen una estructura sencilla, con el identificador de la función, y los parametros necesrios dentro de parentesis, estructura similar al siguiente ejemplo:
funcion(parametros);
De una manera similar, la declaración de funciones definidas pr los usuarios se debe colocar la palabra reservada fun antes del identificador de la función y los parámetros dentro de los paréntesis, estructura que se mantuvo de Perl para declarar funciones en el lenguaje, esto sin la necesidad de definir el tipo que retornara la función ya que Perl es un lenguaje no tipado, lo cual lo hace muy cómodo de usar para solucinar diferentes tipos de programas sin necesidad de preocuparse por tipos, tambi'en las funciones propias del lenguaje permiten realizar diferentes operaciones con arreglos de una manera muy sencilla y entendible para el usuario.

4 Herramientas utilizadas

Como lenguaje de desarrollo de MicroPerl se utilizó Java.
Para el generador del analizador léxico del lenguaje se utilizó el programa JFlex Version 1.6.1, el cual según la pagina oficial de JFlex, es una herramienta que permita generar analizadores léxicos, conocidos como scanner, para el lenguaje Java. El analizador léxico generado por JFlex esta basado en un autómata finito determinista (DFA), realizando el análisis de una manera más eficiente en su procesamiento. Para el generador del analizador sintáctico se utilizó el programa CUP Versión 0.11a, CUP (Construction of Useful Parsers) es una herramienta que permite generar analizadores sintácticos o parsers para un lenguaje determinado desarrollado por C. Scott Ananian, Frank Flannery, Dan Wang, Andrew W. Appel y Michael Petter (www2.cs.tum.edu,S.F). El generador CUP funciona estableciendo una gramática, con sus terminales y no terminales, así como el conjunto de reglas a utilizar para generar el analizador sintáctico, en caso de no haber ciclos , el analizador se genera correctamente, en caso contrario retorna un error por posibles ciclos infinitos en la gramática generada. Con los dos programas anteriores fue posible la construcción de los analizadores léxicos

y semánticos, utilizando el alfabeto previamente definido y la gramática BNF en la cual se definieron las reglas del lenguaje, en cuanto al analizador semántico y el generador de código fue programado por los desarrolladores del equipo, obteniendo así un programa capaz de ejecutar instrucciones de MicroPerl.

5 Referencias bibliográficas

1. Perl Tutorial (s.f.). TutorialsPoint, Simple easy learning. Obtenido de: http://www.tutorialspoint.com/perl/

2. Real Academia Española. (s.f). Diccionario de la lengua española. Definición de lenguaje. Consultado en: http://lema.rae.es/drae/?val=lenguaje

3. ecured(s.f). Lenguaje de Programación. Recuperado de: http://www.ecured.cu/index.php/Lenguaje de_Programación/

4. Real Academia Española. (s.f). Diccionario de la lengua española. Definición de gramática. Consultado en: http://lema.rae.es/drae/?val=gramatica

5. Alvarez, Gloria.(s.f.). Compiladores: Análisis Sintáctico. Obtenido de: http://cic.javerianacali.edu.co/wiki/lib/exe/fetch.php?media=materias:compi:comp_sesion9_2008-1.pdf

6. BNF and EBNF: What are they and how do they work? (s. f.). Obtenido de: http://www.garshol.priv.no/download/text/bnf.html#id1.2 .

7. Serna, Eduardo.(s.f.). Compiladores. Obtenido de: http://www.paginasprodigy.com/edserna/cursos/compilador/notas/Notas2.pdf

8. Louden, Kenneth.(s.f). Construccion de Compiladores. Obtenido de: http://librosysolucionarios.net/construccion-de-compiladores-principios-y-practica-1ra-edicion-kenneth-louden/

9. Definicion.de.(s.f.). Definición de semántica. Obtenido de: http://definicion.de/semantica/

10. Capítulo 5. Análisis semántico.(s.f.). Obtenido de: http://arantxa.ii.uam.es/ãlfonsec/docs/compila5.htm

11. Intérpretes y Diseño de Lenguajes de Programación.(s.f.). Obtenido de: http://di002.edv.uniovi.es/ labra /FTP/Interpretes.pdf

12. Definicion.de.(s.f.). Definición de Java. Obtenido de: http://definicion.de/java/

13. NetBeans IDE.(s.f.). The Smarter and Faster Way to Code. Consultado en: https://netbeans.org/

14. JFLEX. (s.f) JFLEX. Obtenido de: http://jflex.de/

15. CUP. (s.f). CUP Project. Obtenido de: http://www2.cs.tum.edu/projects/cup/

6 Comentarios y Observaciones

- El proyecto asignado en el curso de Compiladores e Interpretes, consistió en el desarrollo de un Interprete en el caso del lenguaje de MicroPerl, resulto muy útil para comprender el desarrollo de los compiladores e interpretes actuales, incluyendo sus 3 partes principales: el analizador léxico o scanner, analizador sintáctico o parser y el analizador semántico, los cuales utilizando estas tres parte de manera correcta y de acuerdo a una gramatica definida correctamente para el lenguaje, se puede generar un compilador completo, capaz de realizar las labores que ejecutan las computadoras en la actualidad, además fue muy importante conocer las estructura interna de un interprete o compilador para comprender mejor el funcionamiento de los lenguajes en la actualidad.
- Las instrucciones implementadas en MicroPerl, son instrucciones para realizar programas básicos y de poca complejidad. El lenguaje fue desarrollado con propósitos académicos y se permite su estudio, modificaciones y utilización del mismo. con la condición de brindar los créditos correspondientes a los desarrolladores del lenguaje MicroPerl.

7 Trabajo futuro

- MicroPerl es un lenguaje que se basa en Perl, MicroPerl tiene un conjunto limitado de instrucciones, por lo cual, más adelante se planea seguir con el proyecto, implementando otras funcionalidades que caracterizan a Perl, por ejemplo el manejo de tipo Hash, el cual por razones de diseño no se incluyeron en esta primera parte del desarrollo del lenguaje, así como el manejo amplio que posee Perl para manejar cadenas de caracteres y el cual el lenguaje MicroPerl tiene un escaso manejo de operaciones con cadenas por lo que se espera que más adelante mejor estas caracteristicas en el lenguaje. También se esperan corregir algunos errores que se presenten con el tiempo de uso y que no fueron contemplados en las primeras etapas de desarrollo, esto con el fin de construir un lenguaje más robusto y completo con las facilidades y capacidades que ofrecen el lenguaje Perl.