Desarrollo de compiladores con software libre
FLEX & BISON

Introducción

En el mundo del software libre es común el uso de intérpretes y compiladores, tales como PERL, PHYTON, FPC, MONO, GCC, entre otros. Muchas de las tareas que se realizan en la computadora son el resultado del análisis de una entrada en especial, tomemos por ejemplo el servidor X, este lee un archivo de configuración en un formato en especial, donde indicamos la resolución y dispositivos de entrada, solo por mencionar algunos. El propósito principal de este tutorial es dar a conocer las metodologías y las herramientas que permitan construir compiladores, interpretes, impresoras estéticas, traductores dirigidos por sintaxis, correctores ortográficos entre otras muchas aplicaciones más, haciendo uso de software libre.

¿QUÉ ES UN COMPILADOR?

El compilador es un programa que se encarga de la traducción global del programa realizado por el usuario. Esta operación recibe el nombre de compilación. El programa es traducido completamente antes de que se ejecute, por lo que la ejecución se realiza rápidamente, ya que se encuentra en un formato fácil de leer para la computadora.

¿QUÉ ES UN INTÉRPRETE?

El intérprete por el contrario lleva a cabo una traducción inmediata en el momento de la ejecución, es decir, irá ejecutando las instrucciones una a una haciendo que el proceso requiera un periodo de tiempo sensiblemente mayor del que necesitaría un compilador. Los intérpretes son usados para traducir programas de alta dificultad de implementación, en estos casos, las órdenes a traducir son de tal complejidad que no merece la pena crear un compilador ya que este también tendría que ser de una complejidad por encima de lo normal.

Lenguaje & Autómatas

Es necesario comprender los fundamentos de la teoría de lenguajes para poder comprender como es que funciona un compilador, una vez que comprendamos como es que funciona un compilador, esto nos permitirá poder construir uno.

CONCEPTOS BÁSICOS

Iniciaremos con los conceptos básicos, tales como símbolo, alfabeto, cadena, y lenguaje. Estos conceptos deben ser estudiados como entidades matemáticas dónde un lenguaje formal es un conjunto de palabras o cadenas formadas por símbolos de un alfabeto dado. Un lenguaje puede ser infinito, aunque el alfabeto debe ser siempre finito. Esto nos ayuda a definir los llamados lenguajes regulares y por ende las expresiones regulares. El concepto de expresión regular tomara forma al analizar varios ejemplos de su aplicación, para esto se usaran varios programas de software libre, como EMACS, BASH y PERL.

AUTÓMATAS

El autómata será usado como herramienta didáctica para comprender el como se puede identificar si una cadena pertenece o no a un lenguaje. El autómata será visto como un grafo y su representación como matriz, estos conceptos serán planteados con la ayuda de un programa, escrito en ANSI C, que reconoce patrones en un archivo de texto en formato ASCII, reconociendo patrones de fecha, hora, IP y URL. Una vez que este programa sea explicado y analizado a fondo el asistente será capaz de comprender la importancia de la teoría computacional y la aplicación directa de la misma.

Conociendo FLEX

Se iniciara, con una breve introducción de la historia de este proyecto y su importancia como herramienta para la construcción de software libre. Una vez comprendida la utilidad de este programa, se procederá a definir el formato de entrada, para crear un analizador léxico, para esto se hará uso de los conceptos de expresión regular vistos con anterioridad, dando continuidad al estudio de creación de compiladores, en este punto se realizara una impresora estética, un programa que resalta en colores el código fuente en Pascal y otro mas para el lenguaje C. A continuación se plantearan diferentes aplicaciones de los analizadores léxicos y la necesidad del análisis sintáctico y el uso de las gramáticas. Estos programas serán implementados en C y C++.

Gramáticas Regulares

Una gramática regular (GR) es la cuarteta G = (N, Sigma, S, P), donde Sigma es un alfabeto, N es la colección de reglas de sustitución, también llamadas producciones. Esto se ilustrara con ejemplos de gramáticas de lenguajes tales como el español y el ingles. Una vez que este concepto sea comprendido se hará uso de autómatas para ilustrar de forma grafica, se hablara de los algoritmos que permiten optimizar las gramáticas y su importancia en la construcción de compiladores.

Conociendo BISON

Una vez que se explorado la importancia del análisis sintáctico y el funcionamiento del mismo, el asistente se iniciara en el uso de BISON para generar analizadores sintácticos, para esto, se construirá una calculadora haciendo uso de BISON. Se hablara de los errores comunes que suelen presentarse cuando se trabaja con este programa y las soluciones que se pueden aplicar a los mismos.

Mi primer compilador

Dado que en este punto ya se han explorado la mayoría de los tópicos y herramientas disponibles para la creación de compiladores, se procederá a crear un pequeño compilador, basado en la maquina de Von Neuman, se definirán las instrucciones básicas de esta maquina. Con este compilador se estudiaran las técnicas de recuperación de errores, la comprobación de tipos, la generación de código intermedio y finalmente la generación de código objeto, el enlazado con librerías y la ejecución.