



NOMBRE: KATHERINE REMACHE

ÁREA ACADÉMICA: TÉCNICA

ESCUELA: ELECTRÓNICA Y TELECOMUNICACIONES

NOMBRE DE LA MATERIA: METODOLOGÍA DE LA
PROGRAMACIÓN

PROFESOR: Ing. PATRICIO PUCHAICELA

SEMESTRE: PRIMERO PARALELO “A”

TIPO DE MATERIA: GENÉRICAS DE CARRERA

AÑO LECTIVO: 2007-2008

INTRODUCCIÓN

LENGUAJES DE PROGRAMACIÓN

Un lenguaje de programación es un lenguaje que puede ser utilizado para controlar el comportamiento de una máquina o computadora. Consiste en un conjunto de reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos, respectivamente.

Un lenguaje de programación permite a uno o más programadores especificar de manera precisa: sobre qué datos una computadora debe operar, cómo deben ser estos almacenados y transmitidos y qué acciones debe tomar bajo una variada gama de circunstancias. Todo esto, a través de un lenguaje que intenta estar relativamente próximo al lenguaje humano o natural, tal como sucede con el lenguaje Léxico.

Los lenguajes de programación son herramientas que nos permiten crear programas y software. Entre ellos tenemos Delphi, Visual Basic, Pascal, Java, etc.

Un lenguaje de programación de una computadora se conoce como código de máquina o lenguaje de máquina, son verdaderamente difíciles de entender para una persona, ya que están compuestos de códigos numéricos sin sentido nemotécnico.

Generalmente la traducción de las instrucciones a un código que comprende la computadora debe ser completamente sistemática. Normalmente es la computadora la que realiza la traducción.

Una computadora funciona bajo control de un programa el cual debe estar almacenado en la unidad de memoria; tales como el disco duro.

Estos lenguajes codificados en una computadora específica no podrán ser ejecutados en otra computadora diferente, para que estos programas funcionen para diferentes

computadoras hay que realizar una versión para cada una de ellas, lo que implica el aumento del costo de desarrollo.

Una característica de los lenguajes de programación es que más de un programador puedan tener un conjunto común de instrucciones, que puedan ser comprendidas entre ellos para realizar la construcción del programa de forma colaborativa.

Los procesadores usados en las computadoras son capaces de entender y actuar según lo indican programas escritos en un lenguaje fijo llamado lenguaje de máquina. Todo programa escrito en otro lenguaje puede ser ejecutado de dos maneras:

- Mediante un programa que va adaptando las instrucciones conforme son encontradas. A este proceso se lo llama interpretar y a los programas que lo hacen se los conoce como intérpretes.
- Traduciendo este programa al programa equivalente escrito en lenguaje de máquina. A ese proceso se lo llama compilar y al traductor se lo conoce como compilador.

Los lenguajes de programación pueden clasificarse de acuerdo a su semejanza con el lenguaje máquina o a su semejanza con el lenguaje humano (generalmente inglés).

Los lenguajes que tiene mayor semejanza con el lenguaje humano se les llama lenguajes de alto nivel, mientras que los lenguajes más parecidos al lenguaje de máquina son conocidos como de bajo nivel.

Entre los lenguajes de bajo nivel se encuentra el lenguaje ensamblador.

Algunos ejemplos de lenguajes de alto nivel son:

- Algol
- Basic
- C
- Cobol
- Fortran
- Modula 2
- Pascal
- Prolog

CLASIFICACIÓN DE LOS LENGUAJES DE PROGRAMACIÓN

Los lenguajes de programación se determinan según el nivel de abstracción, Según la forma de ejecución y Según el paradigma de programación que poseen cada uno de ellos y esos pueden ser:

SEGÚN SU NIVEL DE ABSTRACCIÓN

LENGUAJES DE BAJO NIVEL

Vistos a muy bajo nivel, los microprocesadores procesan exclusivamente señales electrónicas binarias. Dar una instrucción a un microprocesador supone en realidad enviar series de unos y ceros espaciadas en el tiempo de una forma determinada. Esta secuencia de señales se denomina código máquina. El código representa normalmente datos y números e instrucciones para manipularlos. Un modo más fácil de comprender el código máquina es dando a cada instrucción un mnemónico, como por ejemplo

STORE, ADD o JUMP. Esta abstracción da como resultado el ensamblador, un lenguaje de muy bajo nivel que es específico de cada microprocesador, ya que al programar en ensamblador se trabaja con los registros de memoria de la computadora de forma directa.

Los lenguajes de bajo nivel permiten crear programas muy rápidos, pero que son a menudo difíciles de aprender. Más importante es el hecho de que los programas escritos en un bajo nivel son prácticamente específicos para cada procesador. Si se quiere ejecutar el programa en otra máquina con otra tecnología, será necesario reescribir el programa desde el principio.

Lenguaje máquina

El lenguaje propio del ordenador, basado en el sistema binario, o código máquina, resulta difícil de utilizar para las personas. El programador debe introducir todos y cada uno de los comandos y datos en forma binaria, y una operación sencilla como comparar el contenido de un registro con los datos situados en una ubicación del chip de memoria puede tener el siguiente formato: 11001010 00010111 11110101 00101011. La programación en lenguaje máquina es una tarea tan tediosa y consume tanto tiempo que muy raras veces lo que se ahorra en la ejecución del programa justifica los días o semanas que se han necesitado para escribir el mismo.

Lenguaje ensamblador

Uno de los métodos inventados por los programadores para reducir y simplificar el proceso es la denominada programación con lenguaje ensamblador.

El lenguaje máquina es el que la computadora puede ejecutar directamente; sin embargo; no puede llamarse propiamente así a causa de que no está formado por símbolos o signos, como cualquier otro lenguaje, si no que está compuesto por cantidades numéricas expresadas en base 16 o hexadecimal (que permiten hacer más compacto el lenguaje binario que entiende la máquina). De ello se deduce que programar directamente en lenguaje máquina resulta muy complicado. Para evitar esta dificultad existe el lenguaje ensamblador, que es un lenguaje de programación muy cercano al lenguaje máquina se realice mediante muy pocos pasos, ya que sus sentencias conservan la estructura sintáctica del lenguaje máquina.

Pero un lenguaje ensamblador no deja de ser un lenguaje simbólico, el de más bajo nivel. Por eso, dado un programa escrito en ensamblador, para obtener el programa correspondiente en lenguaje máquina, será preciso un proceso de traducción.

El programa que hace la traducción se llama programa ensamblador o, simplemente, ensamblador; su función es muy parecida a la de un compilador, sólo que este traduce lenguajes simbólicos de alto nivel, mientras que el lenguaje ensamblador traduce los de bajo nivel, muy cercanos al lenguaje máquina.

Sin embargo, el lenguaje ensamblador puede utilizarse con un solo tipo de chip de CPU o microprocesador. Los programadores, que dedicaron tanto tiempo y esfuerzo al aprendizaje de la programación de un ordenador, se veían obligados a aprender un nuevo estilo de programación cada vez que trabajaban con otra máquina. Lo que se necesitaba era un método abreviado en el que un enunciado simbólico pudiera representar una secuencia de numerosas instrucciones en lenguaje máquina, y un método que permitiera que el mismo programa pudiera ejecutarse en varios tipos de máquinas. Estas necesidades llevaron al desarrollo de lenguajes de alto nivel.

LENGUAJES DE MEDIO NIVEL

Hay lenguajes de programación que son considerados por algunos expertos como lenguajes de medio nivel (como es el caso del lenguaje C) al tener ciertas características que los acercan a los lenguajes de bajo nivel pero teniendo, al mismo tiempo, ciertas cualidades que lo hacen un lenguaje más cercano al humano y, por tanto, de alto nivel.

LENGUAJES DE ALTO NIVEL

Por lo general se piensa que los ordenadores son máquinas que realizan tareas de cálculos o procesamiento de textos. La descripción anterior es sólo una forma muy esquemática de ver una computadora. Hay un alto nivel de abstracción entre lo que se pide a la computadora y lo que realmente comprende. Existe también una relación compleja entre los lenguajes de alto nivel y el código máquina.

Los lenguajes de alto nivel son normalmente fáciles de aprender porque están formados por elementos de lenguajes naturales, como el inglés del tipo LIST, PRINT u OPEN como comandos que representan una secuencia de decenas o de centenas de instrucciones en lenguaje máquina. En BASIC, el lenguaje de alto nivel más conocido, los comandos como 'IF CONTADOR = 10 THEN STOP' pueden utilizarse para pedir a la computadora que pare si CONTADOR es igual a 10. Por desgracia para muchas personas esta forma de trabajar es un poco frustrante, dado que a pesar de que las computadoras parecen comprender un lenguaje natural, lo hacen en realidad de una forma rígida y sistemática.

Los comandos se introducen desde el teclado, desde un programa residente en la memoria o desde un dispositivo de almacenamiento, y son interceptados por un programa que los traduce a instrucciones en lenguaje máquina.

SEGÚN LA FORMA DE EJECUCIÓN

COMPILADOR



Compilador es un programa informático capaz de generar aplicaciones que sean directamente utilizables en un ordenador o computadora, recibe un programa fuente y devuelve un programa objeto, o sea traduce el código fuente de un programa en lenguaje de alto nivel, a otro lenguaje de nivel inferior (lenguaje máquina, o código binario), los compiladores traducen un programa íntegro a lenguaje máquina antes de su ejecución, por lo cual se ejecutan con tanta rapidez como si hubiesen sido escritos directamente en lenguaje máquina. Este proceso de traducción se conoce como compilación.

Al usar un lenguaje compilado (como lo son los lenguajes del popular Visual Studio de Microsoft), el programa desarrollado nunca se ejecuta mientras haya errores, sino hasta que luego de haber compilado el programa, ya no aparecen errores en el código.

Éstos, como los programas ensambladores avanzados, pueden generar muchas líneas de código de máquina por cada proposición del programa fuente. Se requiere una corrida de compilación antes de procesar los datos de un problema.

De esta manera un programador puede diseñar un programa en un lenguaje mucho más cercano a como piensa un ser humano, para luego compilarlo a un programa más manejable por una computadora.

En un compilador hay que distinguir tres lenguajes diferentes:

- El de los programas de partida (LA)
- El de los programas equivalentes traducidos (LB), normalmente el lenguaje de máquina.
- El lenguaje en que está escrito el propio compilador (LC), que puede ser igual o diferente a uno de los otros dos.

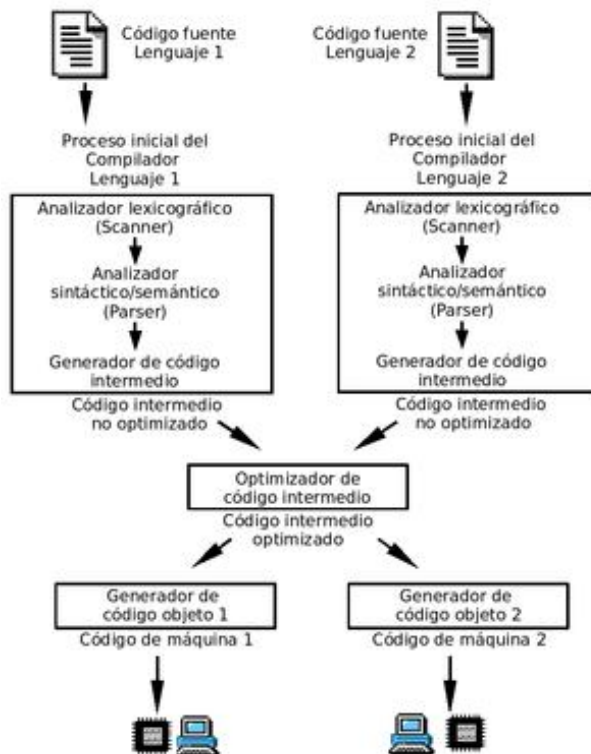


Diagrama a bloques de la operación de un buen compilador.

HISTORIA

En 1946 se desarrolló el primer ordenador digital. En un principio, estas máquinas ejecutaban instrucciones consistentes en códigos numéricos que señalaban a los circuitos de la máquina los estados correspondientes a cada operación, lo que se denominó lenguaje máquina.

Pronto los primeros usuarios de estos ordenadores descubrieron la ventaja de escribir sus programas mediante claves más fáciles de recordar que esos códigos; al final, todas esas claves juntas se traducían manualmente a lenguaje máquina. Estas claves constituyen los llamados lenguajes ensambladores.

A pesar de todo, el lenguaje ensamblador seguía siendo el de una máquina, pero más fácil de manejar.

En las primeras épocas de la informática, el software de los compiladores era considerado como uno de los más complejos existentes.

Los trabajos de investigación se orientaron hacia la creación de un lenguaje que expresara las distintas acciones a realizar de una manera lo más sencilla posible para el hombre. El primer compilador fue escrito por Grace Hopper, en 1952 para el lenguaje de programación A-0, un programa que traduce las instrucciones con palabras en inglés al lenguaje máquina de una computadora

Los primeros compiladores se realizaron programándolos directamente en lenguaje máquina o en ensamblador. Una vez que se dispone de un compilador, se pueden escribir nuevas versiones del compilador (u otros compiladores distintos) en el lenguaje que compila ese compilador.

En 1950 John Backus dirigió una investigación en IBM sobre un lenguaje algebraico. En 1954 se empezó a desarrollar un lenguaje que permitía escribir fórmulas matemáticas de manera traducible por un ordenador; le llamaron FORTRAN (FORmulae TRANslator). Fue el primer lenguaje de alto nivel y se introdujo en 1957 para el uso de la computadora IBM modelo 704.

Surgió así por primera vez el concepto de un traductor como un programa que traducía un lenguaje de alto nivel a otro lenguaje de bajo nivel, se emplea el término compilador.

La tarea de realizar un compilador no fue fácil. El primer compilador de FORTRAN tardó 18 años-persona en realizarse y era muy sencillo, estaba muy influenciado por la máquina objeto en la que iba a ser implementado. Como un ejemplo de ello tenemos el hecho de que los espacios en blanco fuesen ignorados, debido a que el periférico que se utilizaba como entrada de programas (una lectora de tarjetas perforadas) no contaba correctamente los espacios en blanco.

El primer compilador auto contenido, es decir, capaz de compilar su propio código fuente fue el creado para Lisp por Hart y Levin en el MIT en 1962. Desde 1970 se ha convertido en una práctica común escribir el compilador en un mismo lenguaje que este compila, aunque Pascal y C han sido alternativas muy usadas.

El primer compilador creado para un lenguaje tiene que o bien ser compilado en por un compilador escrito en otro lenguaje o bien compilado al ejecutar el compilador en un interprete. Actualmente existen herramientas que facilitan la tarea de escribir compiladores ó intérpretes informáticos. Estas herramientas permiten generar el esqueleto del analizador sintáctico a partir de una definición formal del lenguaje de partida, especificada normalmente mediante una gramática formal y barata, dejando

únicamente al programador del compilador la tarea de programar las acciones semánticas asociadas.

Comparando su actuación con la de un ser humano, un compilador equivale a un traductor profesional que, a partir de un texto, prepara otro independiente traducido a otra lengua, mientras que un intérprete corresponde al intérprete humano, traduce de viva voz las palabras que oye, sin dejar constancia por escrito.

PARTES DE UN COMPILADOR:

Normalmente los compiladores están divididos en dos partes:

- Front End: es la parte que analiza el código fuente, comprueba su validez, genera el árbol de derivación y rellena los valores de la tabla de símbolos. Esta parte suele ser independiente de la plataforma o sistema para el cual se vaya a compilar.

Los front ends varían internamente, teniendo que producir árboles que puedan ser manejados por el back end. Todos los analizadores son analizadores recursivos descendentes y fueron escritos manualmente, no generados automáticamente.

Hasta hace poco, el árbol de representación de programa no era totalmente independiente del procesador para el que se quería generar el código.

- Back End: es la parte que genera el código máquina, específico de una plataforma, a partir de los resultados de la fase de análisis, realizada por el Front End.

Esta división permite que el mismo Back End se utilice para generar el código máquina de varios lenguajes de programación distintos y que el mismo Front End que sirve para analizar el código fuente de un lenguaje de programación concreto, sirva para generar el código máquina en varias plataformas distintas.

El código que genera el Back End normalmente no se puede ejecutar directamente, sino que necesita ser enlazado por un programa enlazador (linker).

El comportamiento del back end está parcialmente especificado por el preprocesador de macros específicas a la arquitectura objetivo, por ejemplo para definir la posición de los bits más significativos, tamaño de palabra, convención para llamadas, etc. El back end utiliza éstas para la generación de RTL, aunque en GCC éste es independiente del procesador, la secuencia inicial de instrucciones abstractas es adaptada a la arquitectura objetivo.

TIPOS DE COMPILADORES:

Esta taxonomía de los tipos de compiladores no es excluyente, por lo que puede haber compiladores que se adscriban a varias categorías:

- Compiladores cruzados: generan código para un sistema distinto del que están funcionando.
- Compiladores optimizadores: realizan cambios en el código para mejorar su eficiencia, pero manteniendo la funcionalidad del programa original.
- Compiladores de una sola pasada: generan el código máquina a partir de una única lectura del código fuente.
- Compiladores de varias pasadas: necesitan leer el código fuente varias veces antes de poder producir el código máquina.

- Compiladores JIT (Just In Time): forman parte de un intérprete y compilan partes del código según se necesitan.

INTERPRETADOR

En informática, un intérprete es un programa capaz de analizar y ejecutar otros programas, escritos en un lenguaje de alto nivel. Los intérpretes suelen contraponerse a los compiladores, ya que mientras que los segundos se encargan de traducir un programa desde su descripción en un lenguaje de programación al código máquina del sistema destino, estos sólo realizan la traducción a medida que sea necesario y normalmente, no guardan el resultado de dicha traducción.

La ventaja del proceso intérprete es que no necesita de dos fases para ejecutar el programa, sin embargo su inconveniente es que la velocidad de ejecución es más lenta, que los compilados ya que debe analizar e interpretar las instrucciones contenidas en el programa fuente, pero a cambio son más flexibles como entornos de programación y depuración y permiten ofrecer al programa interpretado un entorno del propio intérprete (lo que se conoce comúnmente como máquina virtual). Por ejemplo, durante el procesamiento repetitivo de los pasos de un ciclo, cada instrucción del ciclo tendrá que volver a ser interpretado cada vez que se ejecute el ciclo, lo cual hace que el programa sea más lento en tiempo de ejecución (porque se va revisando el código en tiempo de ejecución) pero más rápido en tiempo de diseño (porque no se tiene que estar compilando a cada momento el código completo).

El intérprete elimina la necesidad de realizar una corrida de compilación después de cada modificación del programa cuando se quiere agregar funciones o corregir errores.

Perl, PHP, Javascript, Logo, ASP (hasta la versión 3) y Python son ejemplos de lenguajes que son normalmente interpretados en vez de compilados.

Lenguajes interpretados de uso común

Los lenguajes que suelen ser interpretados más famosos en la actualidad son (en orden alfabético):

- ActionScript
- ASP (hasta la versión 3)
- Bash
- Basic4GL (Basic para OpenGL. Permite generar ejecutables Windows completos)
- BeanShell (Java en lenguaje de Scripts)
- Inform
- IO (es un lenguaje reciente -2002-)
- JavaScript (todas las plataformas)
- Logo (Linux, Windows y Mac)
- Lisp
- Lua
- Lush (Lisp para gráficos Linux)
- NWNScript (empleado en el videojuego Neverwinter Nights y Neverwinter Nights 2)
- Perl (Mundo Unix y en general para todas las demás plataformas)
- PHP
- Pike (es el más rápido de todos, al estar escrito en gran parte en código nativo)
- Python (todas las plataformas)

- REXX y variantes como Object REXX (todas las plataformas, en especial: OS/2 / AmigaOS)
- Ruby
- TCL
- VBScript (Microsoft Windows)

TRADUCTORES

La computadora solo puede ejecutar instrucciones escritas en un lenguaje formado por secuencias de cero y unos, al que normalmente se denomina lenguaje máquina. Por ello, cualquier lenguaje de programación que no sea lenguaje máquina necesitara un proceso de traducción. Este proceso lo llevan a cabo los intérpretes y los compiladores. Un traductor es un programa que recibe como entrada código escrito en un cierto lenguaje y produce como salida código en otro lenguaje. Por ejemplo pasar de español a inglés, de java a c#, etc.

Con un intérprete, los programas que repiten un ciclo para volver a ejecutar parte de sus instrucciones, reinterpretan la misma instrucción cada vez que aparece. Por consiguiente, los programas interpretados se ejecutan con mucha mayor lentitud que los programas en lenguaje máquina.

Generalmente el lenguaje de entrada es de más alto nivel que el de salida. Ejemplos de traductores son los ensambladores y los compiladores.

Un ensamblador es un programa que traduce de un lenguaje ensamblador a lenguaje máquina, mientras que un compilador es un programa que traduce de un lenguaje de alto nivel a un lenguaje de bajo nivel o a lenguaje máquina.

Un traductor es un programa que toma el texto escrito en un lenguaje (el lenguaje

fuente) y lo convierte en el texto equivalente en un segundo lenguaje (el lenguaje destino u objeto).

Si la fuente es un lenguaje de alto nivel y si el objetivo es un lenguaje de ensamblaje de bajo nivel o de máquina, el traductor es un compilador.

Bibliografía:

- Aho Alfred, Compiladores principios técnicas y herramientas
- Garrido Alicia, Diseño de Compiladores, 2002
- Microsoft ® Encarta ® 2007. © 1993--2006 Microsoft Corporation. Reservados todos los derechos.
- Let's Build a Compiler. Tutorial de Jack W. Crenshaw sobre cómo hacer un compilador.
- Java a tope: Traductores y Compiladores con Lex/Yacc, JFlex/Cup y JavaCC. Libro básico sobre compiladores.
- ["http://es.wikipedia.org/wiki/Int%C3%A9rprete_inform%C3%A1tico"](http://es.wikipedia.org/wiki/Int%C3%A9rprete_inform%C3%A1tico)
- Wikilibros alberga un libro o manual sobre Fundamentos de programación.
- Mentor Interactivo. Enciclopedia Temática Estudiantil OCEANO.
- Programación en Open Directory Project.
- wikipedia.org/wiki/Compilador - 27k - En caché - Páginas similares
- www.linux10.com.ar/Glosario/terminos/compilador.htm - 6k - En caché - Páginas similares
- wikipedia.org/wiki/Intérprete_informático - 17k - En caché - Páginas similares
- www.alegsa.com.ar/Dic/compilador.php - 17k - En caché - Páginas similares