Procesadores de lenguajes: traductores

Programar en un lenguaje que no es el de la máquina exige traducir.

Intérprete puro

La regla anterior tiene una excepción: los intérpretes puros.

Un intérprete puro del lenguaje X para la máquina M cuyo lenguaje es LM es un programa escrito en (o traducido al) lenguaje LM.

La máquina M más un intérprete puro de un lenguaje X es un procesador capaz de ejecutar instrucciones escritas en lenguaje X.

La *máquina virtual* (ordenador + un intérprete de Pascal) puede ejecutar programas escritos en Pascal sin necesidad de traducción.

Los intérpretes puros son raros. Por tanto, normalmente hay que realizar una o más traducciones para generar un programa en lenguaje LM ejecutable por la máquina M.

Hay varias posibilidades.

Ensamblador

Ensamblador . Si el código está en lenguaje ensamblador, el traductor a LM también se llama ensamblador.

Preprocesador

Preprocesador : Si dos lenguajes A, B son lo suficiente próximos es posible pasar de A a B sin más que operaciones de proceso de texto (buscar y sustituir, incorporar textos etc.). Este fue el caso de C++ que, antes de tener compiladores propios, era preprocesado a C. En este caso hace falta otra traducción de C a LM.

Compilador

Compilador : Su entrada el es código fuente y la salida el código máquina. Esta forma de trabajo hace que tengan las siguientes características:

La *ejecución* de un programa compilado es rápida en comparación con las versiones interpretadas de los programas ya que la ejecución no se exige la etapa de traducción.

Es *difícil encontrar los errores en ejecución* ya que no se puede decir dónde se han producido. Por ello, se utilizan programas correctores (debugers) que ayudan en esta tarea usando diferentes técnicas como la ejecución paso a paso, los puntos de ruptura o la visualización del contenido de ciertas variables, los volcados de memoria etc.

> *NOTA. Algunos compiladores producen como salida código en ensamblador en lugar de código máquina y en un concepto más amplio de compilador la salida puede estar en cualquier otro lenguaje.*
>
> *NOTA. Algunos lenguajes, como java, usan un traductor a código intermedio que es compilado justo en el momento en que se va a ejecutar (Compilador JIT - Just In Time).*

Por razones de productividad, el *código objeto no es ejecutable* sino que hay que enlazarlo con rutinas preexistentes usando un programa que se llama montador de enlaces que produce el código ejecutable.

Intérprete (no puro)

Intérprete. Casi siempre, por motivos de eficiencia, los intérpretes traducen a un código intermedio antes de ejecutar cada instrucción. En este caso realizan las siguientes operaciones:

Leer una instrucción en código fuente

Traducirla a una o más instrucciones en código interno

Ejecutar ese código

Características de los intérpretes:

Son más *lentos* por la necesidad de traducir cada instrucción antes de ejecutarla

Facilitan la detección de errores ya que detienen el programa indicando la línea de código en la que ha "saltado" el error, lo que no quiere decir que esté forzosamente en ella, pero da pistas sobre cual puede ser. En general, son más interactivos.

Exigen menos memoria . Observe que para interpretar un programa sólo hay que cargar en memoria

- el intérprete
- el espacio necesario para los datos
- la instrucción en ejecución.