Estructura de un Traductor

Un traductor se defina como un programa que traduce o convierte desde un texto o programa escrito en un lenguaje fuente hasta un texto o programa equivalente escrito en un lenguaje destino produciendo, si cabe, mensajes de error. Los traductores engloban tanto a los compiladores (en los que el lenguaje destino suele ser código maquina) como a los interpretes (en los que el lenguaje destino está constituido por las acciones atómicas que pueden ejecutar el intérprete). Es importante destacar la velocidad con la que hoy en día se puede construir un compilador.

Existen dos fases en el proceso de traducción las cuales son:

- ✔ Fase de análisis: Comprueba que el programa está escrito conforme a las reglas del lenguaje.
- ✔ Fase de síntesis: Genera el texto equivalente en el lenguaje objeto.

Fase de análisis:

Análisis léxico: es la primera fase de un compilador, consistente en un programa que recibe como entrada el código fuente de otro programa (secuencia de caracteres) y produce una salida compuesta de *tokens* (componentes léxicos) o símbolos. Estos *tokens* sirven para una posterior etapa del proceso de traducción, siendo la entrada para el analizador sintáctico (en inglés *parser*).

Análisis sintáctico: convierte el texto de entrada en otras estructuras (comúnmente árboles), que son más útiles para el posterior análisis y capturan la jerarquía implícita de la entrada.

Análisis semántico: utiliza como entrada el árbol sintáctico detectado por el análisis sintáctico para comprobar restricciones de tipo y otras limitaciones semánticas y preparar la generación de código.

Fase de síntesis:

El código intermedio: se genera para una máquina virtual. Estas máquinas se definen con dos objetivos: Ser lo suficientemente simples como para poder generar código para ellas de manera sencilla, pero con la suficiente riqueza para poder expresar las construcciones del lenguaje fuente.

Optimización de código: es el conjunto de fases de un compilador que transforman un fragmento de código en otro fragmento con un comportamiento equivalente y que se ejecuta de forma más eficiente, es decir, usando menos recursos de cálculo como memoria o tiempo de ejecución

Generación de código: es un nuevo tipo de componente que los desarrolladores de C# pueden escribir y que le permite hacer dos cosas principales: Recuperar un objeto de compilación que representa todo el código de usuario que se está compilando.

Manejador de errores: Se trata en este caso de detectar y corregir errores aparecidos en las transmisiones. El control de errores en el nivel de enlace de datos se basa en la repetición automática (ARQ), que implica la retransmisión de datos en tres casos: tramas dañadas, tramas perdidas y reconocimiento perdido.