Lenguajes de cuarta generación (4GL)

Los lenguajes de cuarta generación son entornos de desarrollo de aplicaciones constituidos por un conjunto de herramientas integradas entre las que se encuentran editores, compiladores, sistemas para el acceso a bases de datos, generadores de informes, generadores de pantallas (modo carácter, interfaces gráficas), etc.

A diferencia de las herramientas CASE, los 4GL se centran fundamentalmente en las fases de construcción e implantación. En este aspecto, una herramienta CASE del tipo L-CASE tendría muchas semejanzas con un 4GL. De hecho, muchas herramientas U-CASE tienen interfaces con un 4GL para completar el ciclo de vida del desarrollo de sistemas.

Los lenguajes que incorporan los 4GL suelen ser mezcla de lenguajes procedurales y no procedurales. La parte procedural se manifiesta en la definición de tipos de constantes, tipos de datos elementales, visibilidad de las variables (locales o globales), sentencias de control de flujo, definición de funciones y procedimientos, etc., mientras que la parte no procedural suele estar basada en el lenguaje SQL (*Structured Query Language*) o, como mínimo, en lenguajes de consulta de bases de datos relacionales.

Con los 4GL se consigue un aumento de productividad gracias a:

- La utilización de funciones preprogramadas.
- El entorno de desarrollo que facilita la realización de determinadas tareas como diseño de pantallas o informes.

Tipos de 4GL

Los 4GL, en función de su relación con un determinado gestor de base de datos, se pueden agrupar de la forma siguiente:

- **Lenguajes que están ligados a una base de datos.** La mayoría de los gestores de bases de datos cuentan con un lenguaje de cuarta generación. Son lenguajes propietarios, lo que quiere decir que sirven únicamente para acceder a esa base de datos en particular. El aprovechamiento de los recursos del gestor es muy alto.

- **Lenguajes que son independientes del gestor de base de datos.** Tienen la capacidad de acceder a diferentes bases de datos, generalmente aquéllas que soportan un estándar común. No son lenguajes propietarios y por tanto no ligan al comprador a ninguna base de datos en particular. La necesidad de utilizar el 4GL siguiendo estrictamente el estándar para asegurar la accesibilidad a diferentes bases de datos impide sacar el máximo provecho de cada una de ellas.

Otra forma de agrupar los 4GL es en función de la naturaleza de su sintaxis:

- **Lenguajes procedurales.** El programa se desarrolla como una secuencia de pasos que la computadora ejecuta para llegar al fin deseado. Los desarrolladores deben codificar los flujos de control de las actividades a realizar, además de las actividades en si.

- **Lenguajes conducidos por eventos.** Permiten a los desarrolladores especificar la ejecución de rutinas asociadas a acciones dadas del usuario, tales como apretar un tecla o mover el ratón, sin tener que codificar cada paso dado para ejecutar dicha acción.

Componentes y funcionalidades de un 4GL

Los principales componentes de un lenguaje de cuarta generación son:

Editor

Donde se escriben las sentencias del lenguaje de programación. Puede contar con:

- Ayuda de tratamiento de textos.
- Facilidades para incorporar el nombre de variables, objetos o funciones.
- Chequeo preliminar de errores de sintaxis.
- Utilidades de selección, copia o movimiento de bloques.
- Posibilidad de deshacer el último cambio.

Compilador

Traduce las sentencias del lenguaje fuente a código binario o a un lenguaje intermedio. Las características más importantes de un compilador son:

- Posibilidad de separar la interpretación del código fuente de la generación del código. Esto permite la ejecución inmediata de una parte del código sin haber generado el fichero ejecutable.
- Gestión avanzada de errores. Recuperación desde un estado erróneo del código para poder continuar con el proceso de interpretación y así detectar el mayor número posible de errores en una única compilación.
- Optimización del código. La traducción del código fuente va acompañada por una optimización del código (en tamaño y/o en rendimiento) a la hora de ejecutar la aplicación.
- Algunos lenguajes incorporan también la posibilidad de utilizar y generar librerías de enlace dinámico (DDL), lo que permite una mayor flexibilidad y capacidad de reutilización
- del código.

Módulo de acceso a base de datos

Incorpora la interfaz con el gestor de base de datos. Facilita toda la comunicación con la base de datos, desde el diseño de las tablas hasta la construcción de sentencias para recuperar información. La mayoría de los 4GLs soporta el lenguaje SQL estándar como lenguaje de acceso a base de datos relacionales, lo que garantiza la portabilidad.

Módulo de ayuda a las pruebas

Hay 4GLs que permiten una ejecución controlada del código para poder aislar un error, con técnicas de ejecución paso a paso, localizando los puntos de parada y permitiendo la modificación del contenido de las variables durante la ejecución.

Generador de informes y pantallas

Los 4GLs incorporan módulos para la construcción rápida de pantallas, ya sea en modo carácter o en modo gráfico. Asimismo, algunos cuentan con un módulo de generación de informes a través de consultas a la base de datos.

Diccionario

Algunos 4GLs cuentan con un diccionario en el que almacenan la información referente a los objetos de la aplicación. Esto facilita la gestión de los objetos generados especialmente para trabajos en grupo.

Gestor de librerías

El gestor de librerías permite:

- La distribución de los objetos por las librerías siguiendo los criterios que se establezcan.
- La localización rápida de los objetos con el fin de analizar el impacto de una modificación o corregir un error.
- La coordinación de los trabajos en equipo.

Módulo de control de versiones

Algunos lenguajes de cuarta generación incorporan facilidades para el control de versiones o tienen interfaz con alguna herramienta del mercado para el control de versiones.

Biblioteca con funciones u objetos reutilizables en la aplicación

La funcionalidad de este tipo de bibliotecas se describe en detalle en el apartado de otras herramientas al hablar de bibliotecas de clases de objetos.

Tendencias tecnológicas y del mercado

La evolución de los 4GLs parece dirigirse hacia:

- Independencia de plataformas *hardware* y *software.* Es importante reseñar la tendencia de algunos 4GLs hacia la generación de lenguaje C, con la ventaja que supone en cuanto a portabilidad.

- Independencia de estructuras de datos y acceso a información distribuida.

- Acceso a objetos distribuidos, lo que permite independizar los recursos que utilizará la aplicación a desarrollar, de la localización física de los mismos.

- Interoperabilidad con herramientas ofimáticas.

- Soporte para diferentes interfaces gráficas de usuario.

- Soporte para diferentes entornos de red.

- La aplicación de forma más extendida del modelo cliente/servidor, tanto en el funcionamiento del propio 4GL como en las aplicaciones generadas.

- Soporte para desarrollo de aplicaciones para Internet e Intranet. Es destacable la tendencia actual a modelos de desarrollo en los cuales la base de datos y la lógica de la aplicación residen bien en un servidor común, bien en distintos servidores, mientras que la presentación de la misma en el cliente se realiza mediante un browser a través de una red Intranet.

- Incorporación de la tecnología de orientación a objetos. Si bien son pocas las herramientas que implementan esta tecnología de forma completa, muchas de ellas si se aprovechan ya de algunas funciones propias de la misma.

- Mayor apertura para la interfaz con herramientas CASE.

- Compatibilidad con otros lenguajes. Algunas herramientas ya son capaces de utilizar módulos generados por otros lenguajes.

- Aplicación de capacidades multimedia.