11.1. INTRODUCCIÓN. ENTORNO DE DESARROLLO CAD.

Un Sistema CAD dispone prácticamente siempre de un entorno de desarrollo[1] con el que poder efectuar **macros de operaciones repetitivas**, procedimientos específicos que se transforman en **nuevas instrucciones de usuario** o conjuntos de procedimientos relacionados entre sí que dan origen a una **aplicación vertical**. Una aplicación vertical se instrumenta en forma de nuevos comandos de usuario que sólo son ejecutables dentro del entorno del sistema CAD, es decir, cuando esta aplicación CAD está en ejecución, nunca directamente desde el Sistema Operativo.

El entorno de desarrollo de un Sistema CAD se compone de **un lenguaje de programación convencional** (Pascal, C, LISP, BASIC, etc.) y **una biblioteca de funciones CAD**, de manera que el programador debe conocer o estudiar los fundamentos generales de programación y complementar ese conocimiento con las prestaciones que posee la biblioteca CAD complementaria, que le permite programar fácilmente procedimientos gráficos específicos. Los fundamentos de una programación convencional se establecen según dos componentes básicas: los **algoritmos** y las **estructuras de datos**; los primeros son flujos ordenados de operaciones que pueden ser ejecutados en un computador; las segundas son formas de organizar la información que se procesa. En conjunción con los algoritmos, las estructuras de datos permiten que los problemas que se programan tengan una solución eficiente.

11.1.1. EXPRESIÓN DE ALGORITMOS.

Un lenguaje de programación de alto nivel permite construir un programa por medio de una serie de procedimientos, que constan de parámetros y sentencias , según un formato genérico del tipo: **procedure-name (parámetros) sentencias . . .**. Una sentencia puede ser de uno de los siguientes tipos:

- **Asignación**, del tipo **variable = valor**, que proporciona a una variable un valor.
- **Condición**, del tipo: **IF condición, THEN sentencia1 [ELSE sentencia2]**, que permite que se realize la sentencia 1 o la sentencia 2 en función del resultado lógico de la condición.
- **Ciclo**, del tipo **WHILE condición DO sentencias …**, o similar, que permite iterar cíclicamente el conjunto de sentencias hasta que deje de satisfacerse la condición.
- **Devolución**, del tipo **RETURN expresión**. Una sentencia de este tipo aparece en un procedimiento de naturaleza:
- **Función**, del tipo **FUNCIÓN-nombre (parámetros) sentencias …** La expresión del return es el segundo miembro de una expresión de asignación como la que se propone 9 líneas más arriba.

---

[1] Tanto AutoCAD como Microstation ofrecen en realidad dos entornos, uno a nivel interpretado y otro compilado. En este libro se hace referencia a AutoLISP y MBE, entornos interpretados en ambas aplicaciones. Los compilados se denominan ADS y MDL, respectivamente, y utilizan C como lenguaje de base. Es recomendable el aprendizaje a partir de los entornos interpretados; prácticamente en su totalidad, los recursos de programación de gráficos son los mismos en unos u otros. La potencia de ADS y MDL se aprecia cuando se necesitan bibliotecas complementarias, fáciles de obtener e integrar cuando se trabaja con C como lenguaje de programación de base.

11.1.2. ESTRUCTURAS DE DATOS.

Los algoritmos entrañan la manipulación de objetos que no son tratables a nivel de máquina, de modo que es preciso que el programador los organice en base a tipos simples de datos que sean directamente representables por el computador; esas organizaciones se denominan Estructuras de Datos. Las más comunes son los conjuntos y los conjuntos ordenados (arrays, listas, listas ligadas, árboles, etc.) Las operaciones fundamentales que se requieren para manipular un conjunto son[2]:

- **Member**, para saber si un elemento pertenece al conjunto.
- **Insert**, para incorporar un elemento al conjunto.
- **Delete**, para eliminar un elemento del conjunto.
- **Find**, para encontrar la posición de un elemento en el conjunto.
- **Union**, para generar un conjunto como union de varios.

Si el conjunto está ordenado, estas operaciones añadidas son importantes:

- **Min**, para obtener el elemento más bajo en la ordenación.
- **Split**, para partir el conjunto en dos complementarios y ordenados.
- **Concatenate**, para generar un conjunto ordenado a partir de dos, también ordenados y que cumplen que cualquier elemento de uno de ellos es menor que cualquier elemento de los del otro.

11.1.3. BIBLIOTECA DE FUNCIONES CAD.

La biblioteca de CAD recoge prácticamente los mismos recursos estudiados en los capítulos anteriores (dibujo de elementos, entrada de datos, atributos, consulta, estructuras jerárquicas, visualización, etc.) en forma de funciones de programación, no de comandos de usuario. Esta circunstancia puede hacer pensar que lo ideal sería entonces que la biblioteca CAD fuera, simplemente, una biblioteca normalizada de funciones gráficas como PHIGS o GKS; en efecto, eso es ideal, pero los constructores de Sistemas CAD no lo ofrecen así por dos razones fundamentales:

1. Porque las bibliotecas CAD están diseñadas a alto nivel, mientras que una biblioteca normalizada de Gráficos por Computador como PHIGS o GKS lo está a medio nivel[3].
2. Porque muchas de las funciones de programación CAD recurren a los propios comandos de usuario, invocados en forma de sentencia o de función.

Las semejanzas conceptuales entre una biblioteca de funciones CAD y una de gráficos por Computador son elevadas pero desde el punto de vista operativo, las funciones CAD de programación no siguen las abstracciones de normas como PHIGS o GKS. Cada constructor de Sistemas CAD ha diseñado su biblioteca de funciones de programación específica.

---

[2] Se mantiene la expresión inglesa porque es casi exclusiva en esta terminología.

[3] Las diferencias se han ido presentando a lo largo de todos los capítulos anteriores, en especial en sus conclusiones.