PARADIGMAS DE PROGRAMACIÓN

Existe una infinidad de definiciones de lo que es un paradigma. Un paradigma es un determinado marco desde el cual miramos el mundo, lo comprendemos, lo interpretamos e intervenimos sobre él. Abarca desde el conjunto de conocimientos científicos que imperan en una época determinada hasta las formas de pensar y de sentir de la gente en un determinado lugar y momento histórico.

Adam Smith define paradigma, en su libro "Los poderes de la mente", como "*un conjunto compartido de suposiciones. Es la manera como percibimos el mundo: agua para el pez. El paradigma nos explica el mundo y nos ayuda a predecir su comportamiento*".

En nuestro contexto, el paradigma debe ser concebido como una forma aceptada de resolver un problema en la ciencia, que más tarde es utilizada como modelo para la investigación y la formación de una teoría. También, el paradigma debe ser concebido como un conjunto de métodos, reglas y generalizaciones utilizadas conjuntamente por aquellos entrenados para realizar el trabajo científico de investigación.

En nuestro contexto, ***los paradigmas de programación nos indican las diversas formas que, a lo largo de la evolución de los lenguajes, han sido aceptadas como estilos para programar y para resolver los problemas por medio de una computadora.***

Se muestran a continuación un resumen de los paradigmas de uso más extendido en programación.

PROGRAMACIÓN POR PROCEDIMIENTOS

Es el paradigma original de programación y quizá todavía el de uso más común. En él, el programador se concentra en el procesamiento, en el algoritmo requerido para llevar a cabo el cómputo deseado.

Los lenguajes apoyan este paradigma proporcionando recursos para pasar argumentos a las funciones y devolviendo valores de las funciones. FORTRAN es

el lenguaje de procedimientos original, Pascal y C son inventos posteriores que siguen la misma idea. La ***programación estructurada*** se considera como el componente principal de la ***programación por procedimientos***.

PROGRAMACIÓN MODULAR

Con los años, en el diseño de programas se dio mayor énfasis al diseño de procedimientos que a la organización de la información. Entre otras cosas esto refleja un aumento en el tamaño de los programas. La programación modular surge como un remedio a esta situación. A menudo se aplica el término módulo a un conjunto de procedimientos afines junto con los datos que manipulan. Así, el paradigma de la programación modular consiste en:

a) Establecer los módulos que se requieren para la resolución de un problema.

b) Dividir el programa de modo que los procedimientos y los datos queden ocultos en módulos.

Este paradigma también se conoce como principio de ocultación de procedimientos y datos. Aunque C++ no se diseño específicamente para desarrollar la programación modular, su concepto de clase proporciona apoyo para el concepto de módulo.

ABSTRACCIÓN DE DATOS

Los lenguajes como ADA y C++ permiten que un usuario defina tipos que se comporten casi de la misma manera que los tipos definidos por el lenguaje. Tales tipos de datos reciben a menudo el nombre de tipos abstractos o tipos definidos por el usuario. El paradigma de programación sobre este tipo de datos consiste en:

a) Establecer las características de los tipos de datos abstractos se desean definir.

b) Proporcionar un conjunto completo de operaciones válidas y útiles para cada tipo de dato.

Cuando no hay necesidad de más de un objeto de un tipo dado, no es necesario este estilo y basta con el estilo de programación de ocultamiento de datos por medio de módulos.

PROGRAMACIÓN ORIENTADA A OBJETOS (OOP)

El problema con la abstracción de datos es que no hay ninguna distinción entre las propiedades generales y las particulares de un conjunto de objetos. Expresar esta distinción y aprovecharla es lo que define a la OOP a través del concepto de herencia. El paradigma de la programación orientada a objetos es, entonces,

a) Definir que clases se desean
b) Proporcionar un conjunto completo de operaciones para cada clase
c) Indicar explícitamente lo que los objetos de la clase tienen en común empleando el concepto de herencia

En algunas áreas las posibilidades de la OOP son enormes. Sin embargo, en otras aplicaciones, como las que usan los tipos aritméticos básicos y los cálculos basados en ellos, se requiere únicamente la abstracción de datos y/o programación por procedimientos, por lo que los recursos necesarios para apoyar la OOP podrían salir sobrando.