Universidad Abierta Interamericana

Campus Virtual UAIOnline

PROGRAMACIÓN I

LECTURA

UNIDAD 6

ESTILOS DE PROGRAMACIÓN

Autor de contenidos:
Nicolás Battaglia

PARADIGMAS DE PROGRAMACIÓN

¿Qué es un paradigma de programación?

Un paradigma de programación representa una particular manera de enfocar los procesos de construcción de un software. El programador al elegir un paradigma, está asumiendo una visión determinada sobre los métodos para la construcción de un programa o subprograma.

Por lo tanto, diferentes paradigmas resultan en diferentes formas de pensar la **solución de problemas** (tenga en cuenta que con la solución de múltiples "problemas" se construye una aplicación) y en diferentes **estilos de programación**

En este Módulo y en esta unidad en particular, presentaremos los paradigmas vigentes y más utilizados en los últimos años. En el Módulo II, nos focalizaremos en uno de ellos: el Paradigma Orientado a Objetos. Centraremos allí la actividad práctica a través de los Laboratorios que procurarán guiarlo/a en la programación en Visual.NET.

En esta unidad esperamos que usted, a través del estudio, alcance las siguientes metas de aprendizaje:

- Conocer los distintos tipos de paradigmas.
- Analizar e identificar las ventajas y desventajas de cada paradigma.
- Relacionar los paradigmas con los lenguajes más próximos a ellos.

A continuación, le presentamos un detalle de los contenidos y actividades que integran esta unidad. Usted deberá ir avanzando en el estudio y profundización de los diferentes temas, realizando las lecturas requeridas y elaborando las actividades propuestas.

1. **Paradigmas de Programación**

1.1 Paradigma lógico

1.2 Paradigma estructurado

1.3 Paradigma orientado a objetos

1.4 Paradigma orientado a eventos

. PARADIGMAS DE PROGRAMACIÓN

En el campo de las Ciencias Sociales el término **paradigma** se refiere al conjunto de prácticas que definen a una disciplina científica en un período determinado de tiempo.

En nuestro campo, el de la informática, un **paradigma de programación** determina la visión y métodos de un analista en la construcción de un sistema. El sujeto elige, entre todos los paradigmas de programación, aquel que cree más eficaz para ese sistema.

Diferentes paradigmas resultan en diferentes estilos de programación y en diferentes formas de pensar la solución de problemas (con la solución de múltiples "problemas" se construye una aplicación).

No obstante estas diferencias, todos tienen en común –en mayor o en menor medida- el uso de los siguientes métodos o procedimientos:

- La **modularidad -** descomposición lógica de un sistema en entidades más pequeñas - contribuye en este sentido al permitir descomponer el problema en unidades discretas más simples, más pequeñas y manejables. También permite el intercambio de las mismas. Dichas unidades o módulos tienen entre sí un grado de *cohesión* y de *acoplamiento*.

- La **recursividad** significa aplicar una función como parte de la definición de esa misma función. El concepto de recursividad va ligado al de *repetición*. Son recursivos aquellos algoritmos que, estando encapsulados dentro de una función, son llamados desde ella misma una y otra vez.

  En contraposición, están los algoritmos iterativos, que hacen uso de bucles **while**, **do-while**, **for**, etc. El caso típico de recursividad es el cálculo del factorial de un número dado.

Le proponemos realizar una revisión de las principales características de los paradigmas de programación más utilizados en nuestro medio.

1.1 PARADIGMA LÓGICO

La Programación Lógica es un paradigma de programación basado, como su nombre lo indica, en la Lógica. Los programas construidos en un **lenguaje lógico** están construidos únicamente por expresiones lógicas, es decir, que son ciertas o falsas, en oposición a una expresión interrogativa (una pregunta) o expresiones imperativas (una orden).

El paradigma lógico es el que comúnmente se utiliza en **Inteligencia Artificial**. Tal como lo señalamos se basa en axiomas, en sentencias sobre las que debe demostrarse su veracidad mediante la búsqueda en una enorme base de datos.

La Inteligencia Artificial es el resultado de implementar en un objeto inanimado las facultades humanas que configuran la inteligencia. El lenguaje mas conocido de este paradigma es el **Prolog**.

1.2 PARADIGMA ESTRUCTURADO

A las características de este paradigma usted las estudió en Programación Estructurada. Le recordamos algunos aspectos fundamentales.

La programación bajo este paradigma divide el proceso en bloques que pueden o no comunicarse entre si. Estos **bloques** son:

Utiliza tres estructuras básicas de control:

- El control secuencial
- El control de selección
- El control iterativo

Trabajar con este paradigma tiene varias ventajas, entre ellas, permite reutilizar el código programado y otorga una mejor compresión de la programación.

Su opuesto es el *paradigma inestructurado* (muy poco usado). Como su nombre lo indica no tiene ninguna estructura, es simplemente un "bloque", como por ejemplo, los archivos batch (.bat).

1.3 PARADIGMA ORIENTADO A OBJETOS

Tal como lo hemos señalado, cada paradigma tiene sus lenguajes específicos. El Paradigma Orientado a Objetos (POO) tiene, entre otros a **Visual.Net**. Este es el lenguaje que utilizaremos para realizar los Laboratorios que iniciarán en la Unidad 3.

Por ello, veamos algunos conceptos básicos contemplando que su ampliación y profundización será motivo de estudio en la asignatura que lleva el mismo nombre.

El POO se basa en la idea de **encapsular estado** y **operaciones** en **objetos**. En general, la programación se resuelve comunicando dichos objetos a través de mensajes (**programación orientada a mensajes**).

Los pilares de este paradigma, son: herencia, encapsulamiento, polimorfismo y abstracción.

Su principal ventaja es la reutilización de códigos y su facilidad para pensar soluciones a determinados problemas.

Los lenguajes que responden a este paradigma son **Simula**, **Smalltalk**, **C++**, **Java**, **Visual Basic .NET**, etc.

Veamos algunos conceptos claves del Paradigma Orientado a Objetos:

Objeto

Un **objeto** es cualquier cosa real o abstracta que posee una estructura que lo define y acciones que lo controlan.

Identidad

Un objeto presenta una **identidad**, un **estado** y un **comportamiento** en un momento dado. La identidad de un objeto le permite ser distinguido de entre otros y esto se da gracias a la *dirección de memoria*. Los objetos son distintos si ocupan distintas direcciones de memoria.

Estado

El **estado** de un objeto es el conjunto de valores concretos que lo caracterizan en un momento dado, como peso, color, precio, etc.

Clase

Una **clase** es una construcción estática que describe comportamientos comunes y atributos, incluyendo en ella, datos y métodos o funciones.

Métodos

Los **métodos** por defectos son el constructor que inicializa valores y el destructor que libera recursos al finalizar la vida útil de una instancia de una clase creada en memoria.

Instancia

A esta **instancia** de una clase se la llama objeto, quien posee identidad, comportamiento y estado fijo o variable especificas a ese objeto. **Instanciación** es la acción por la cual se crean instancias de una clase. Los objetos creados corresponden al tipo de la clase que lo origina

Encontrará estos conceptos desarrollados en la bibliografía cuya lectura le solicitamos en este momento.

1.4 PARADIGMA ORIENTADO A EVENTOS

Evento

Un **evento** es un suceso que ocurre en un sistema. Eventos pueden ser un *click*, *doble click*, mover el mouse, mover una tecla especial, etc.

Tal como su nombre lo indica, este paradigma esta orientado hacia la resolución de eventos provocados por el programa o desde el exterior. La realización de un suceso dependerá de la definición de eventos que realice el programador. Él determinará los eventos a los cuales el programa reaccionará y las acciones que seguirá al presentarse cada uno. Esto se conoce como **manejador de eventos**.

Al ejecutarse un programa así programado, se iniciará con el código escrito y luego, el programa esperará a que ocurra un evento (clic, doble clic, movimiento del Mouse, teclado, etc). Al ocurrir "algo", se ejecutará el código correspondiente al evento que se realizó.

Mientras se realiza el código correspondiente a un evento es normal que el resto del sistema continúe en otro estado distinto sin saber lo que esta ocurriendo en él salvo que el programador haya definido lo contrario mediante el uso de variables o estructuras en el ambiente global, es decir, en el módulo .

Vbasic 6 es un buen ejemplo de programación orientada a eventos por que justamente, no tiene herencia (es decir, no se produce la herencia de los atributos ni los métodos de ningún objeto superior).