Características de la Orientación a Objetos

Sitio: Agencia de Habilidades para el Futuro
Curso: Modelado y Diseño de Software 1º D
Libro: Características de la Orientación a Objetos

Imprimido por: Eduardo Moreno
Día: lunes, 31 de marzo de 2025, 10:06

Tabla de contenidos

1. Introducción

2. Abstracción

3. Polimorfismo

4. Jerarquía

Introducción

Cuando trabajamos con objetos y clases, es decir con UML construyendo los diagramas y con un lenguaje orientado a objetos para la programación, vimos que existe una serie de características que se deben cumplir.

Las características más importantes son:

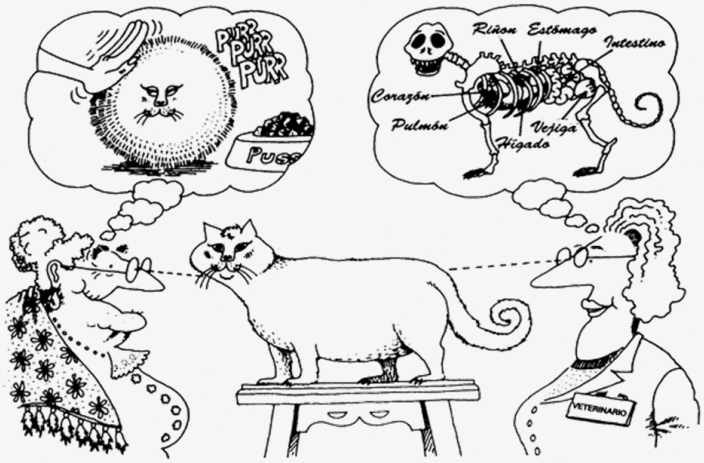
1. Abstracción
2. Polimorfismo
3. Jerarquía

¡Descubramos a qué nos referimos con cada una de ellas!

1. Abstracción

La primera características de la orientación a objetos es la abstracción.

Abstraer es el acto de identificar y utilizar sólo aquellas características pertinentes al propósito actual. Se centra en las características esenciales de algún objeto, en relación a la perspectiva del observador/a.



En la imagen podemos notar que la señora ve al gato como una mascota y solo lo observa como alguien a quien mimar, pero la veterinaria ve al gato como un elemento de su trabajo porque lo que le interesa son las partes que lo componen.

¡Te proponemos un desafío!

Encontrá otro individuo que realice la abstracción del gato. ¿Cuál sería? ¿Qué parte abstraería?

Cuando definimos las características más importantes de un objeto, no nos preocupa cómo se escribirán en el código del programa, simplemente lo definimos de forma general de acuerdo al observador/a.

La herramienta más importante para soportar la abstracción es la clase. Básicamente, una clase es un tipo de dato que agrupa las características comunes de un conjunto de objetos.

Poder ver los objetos del mundo real que deseamos trasladar a nuestros programas, en términos abstractos, resulta de gran utilidad para un buen diseño del software, ya que nos ayuda a comprender mejor el problema y a tener una visión global de todo el conjunto

Por ejemplo:

Si pensamos en una clase Vehículo que agrupa las características comunes de todos ellos, a partir de dicha clase podríamos crear objetos como Coche y Camión. Entonces se dice que Vehículo es una abstracción de Coche y de Camión.

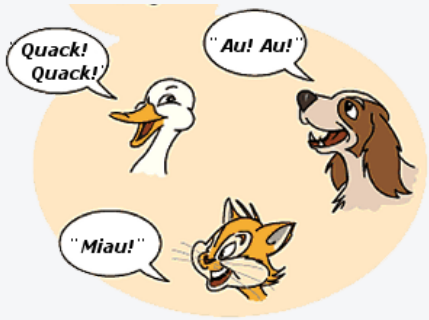
2.- Polimorfismo

Otra de las características a tener en cuenta es el polimorfismo.

Esta propiedad indica la capacidad de que varias clases creadas a partir de una antecesora realicen una misma acción de forma diferente.

Por ejemplo, pensemos en la **clase Animal** y la acción de expresarse.

Nos encontramos que cada tipo de Animal puede hacerlo de manera distinta, los Perros ladran, los Gatos maúllan, etc. Dicho de otra manera, el polimorfismo indica la posibilidad de **tomar un objeto** (de tipo Animal, por ejemplo), e indicarle que realice la acción de expresarse; esta acción será diferente según el tipo de animal del que se trate, tal como se observa en la imagen.



3. Jerarquía

Otra característica a destacar de la orientación a objetos es el concepto de jerarquía.

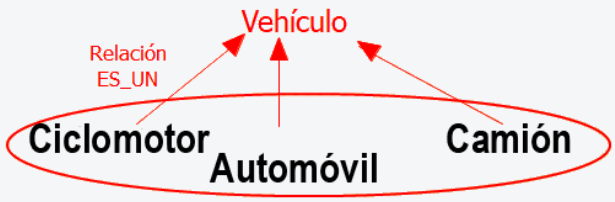
Anteriormente hablábamos sobre las abstracciones y cómo estas definían el comportamiento general de un objeto. En programación orientada a objetos, estas abstracciones pueden ordenarse y clasificarse; a esto se le conoce como **jerarquía** .

Vamos a centrarnos en las dos jerarquías más importantes:

- Relación de clases conocida como **generalización.**
- Relación de partes conocida como **agregación.**

La abstracción de generalización es una técnica utilizada en la programación orientada a objetos que permite, agrupando clases similares, armar en una clase base o "padre", crear clases hijas que heredan las propiedades y métodos de la clase padre. Este tipo de abstracción se lo conoce como **herencia.**

Por ejemplo:



En el ejemplo de la imagen, la clase padre es Vehículo, que tiene sus propiedades, las que son heredadas por las clases hijas que, además, agregan a las del padre, sus propiedades específicas.

Camión tiene como propiedades:

Patente, modelo, potencia CV (heredadas del padre) + nombre empresa (es propia del Camión)

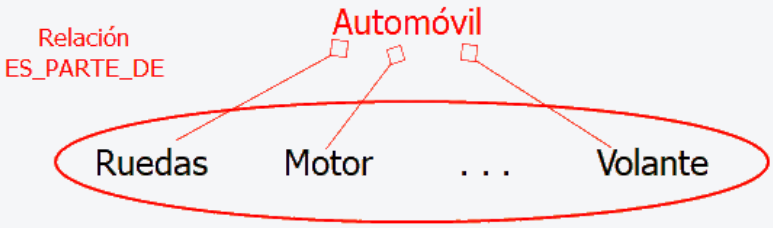
La **generalización** se puede representar visualmente utilizando árboles, donde la clase padre se encuentra en la raíz del árbol y las clases hijas se ubican debajo de ella. Cada clase hija hereda todas las propiedades y métodos de la clase padre y puede agregar sus propias propiedades y métodos.

El uso de la abstracción de generalización en la programación es importante ya que permite

- La reutilización de código.
- Simplifica la estructura de clases.
- Facilita la comprensión de la relación entre las clases.
- Promueve la modularidad y la escalabilidad del código, lo que facilita su mantenimiento y actualización.

La *a*bstracción de **agregación** a grupa conceptos no similares. Ignora las diferencias entre las partes y se concentra en el hecho de que forman parte de un todo.

Define una nueva clase a partir de un conjunto de otras clases que representan sus partes componentes.



Tomando el ejemplo de vehículo, este está formado por los diferentes elementos que lo componen. Las partes que forman el automóvil.

Este tipo de abstracción es muy útil en el diseño y desarrollo de software ya que permite simplificar la complejidad de los sistemas al dividirlos en partes más pequeñas y manejables. Además, la abstracción de agregación facilita la reutilización de código al permitir que diferentes clases sean combinadas para crear nuevas clases sin tener que reescribir el código desde cero.

Es importante mencionar que la abstracción de agregación no es lo mismo que la herencia. La herencia se usa para crear una clase nueva a partir de una clase existente, mientras que la abstracción de agregación se refiere a la creación de una nueva clase a partir de varias clases existentes.

En el ejemplo anterior, la clase vehículo está compuesta por diferentes partes componentes como las ruedas y el motor, pero cada una de estas partes también puede ser utilizada en otros casos. Por ejemplo, la clase rueda puede ser usada en la creación de la clase bicicleta o la clase motocicleta.