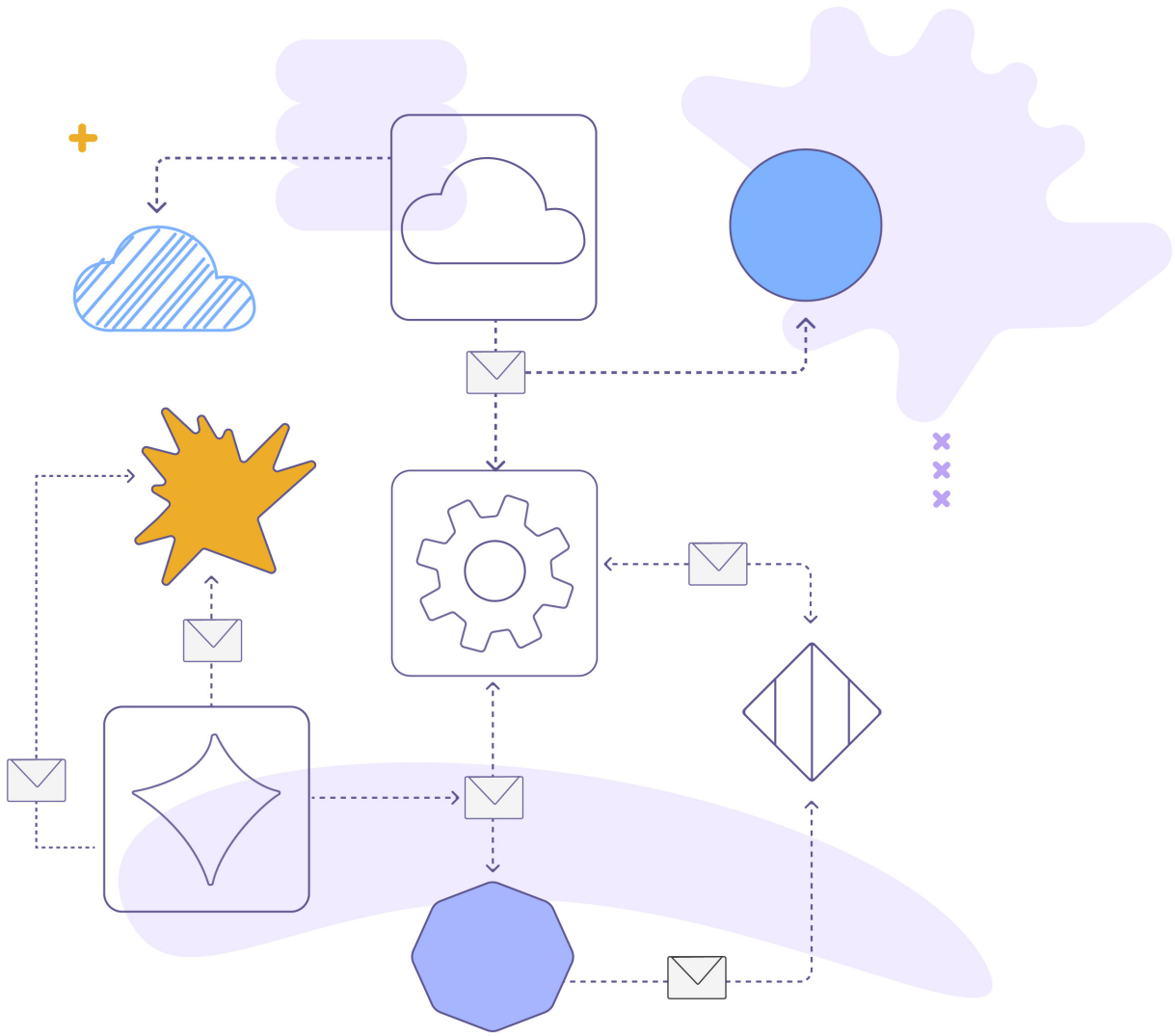
# Capítulo 9

## Análisis y diseño de algoritmos



# Capítulo 9

## Paradigma orientado a objetos

# # Objetivo

En este capítulo aprenderemos los fundamentos del paradigma orientado a objetos como metodología fundamental para la programación actual. Se brindarán las herramientas y el conocimiento para crear soluciones eficaces, flexibles y reutilizables. Se desarrollarán conceptos elementales como las clases, los objetos y los métodos, entre otros.

Al finalizar este capítulo, estarás capacitado para comprender y aplicar los fundamentos de la programación orientada a objetos. Habrás adquirido una comprensión profunda de los beneficios que aporta.

# # Paradigma

En nuestro mundo, el de la programación, un paradigma es una forma de pensar. Es el marco que define cómo se diseñan, estructuran y escriben los programas.

En este sentido, el paradigma orientado a objetos, es un modelo de programación que se basa en la idea de que un programa puede entenderse como un conjunto de objetos que interactúan entre sí para realizar una tarea.

El paradigma orientado a objetos se caracteriza por la encapsulación -capacidad de ocultar los detalles internos-, la herencia -capacidad de crear clases a partir de otras- y el polimorfismo -capacidad de que un objeto se comporte diferente dependiendo el contexto-, conceptos que luego ampliaremos dado que son claves para comprender su dominio.

Partamos de un ejemplo concreto como para tener una idea cercana a qué nos referimos cuando hablamos de objetos o clases.

En la vida real existen diferentes tipos de vehículos, autos, camiones o colectivos. Más allá de sus diferencias todos comparten ciertas características en común, todos tienen ruedas, un motor, una patente o un color determinado. Entonces podemos decir que “Vehículo” representa la clase de objetos que puede agrupar a todos estos. La clase especifica cuáles serán las propiedades comunes que podrían tener.

Por el otro lado, cada vehículo individual será un objeto de esta clase Vehículo. En nuestro

sistema, los vehículos individuales serán las instancias con sus características específicas. Por ejemplo un auto Fiat, modelo Punto, color rojo con la patente AA-002-AA.

Si bien podemos llegar a tener objetos de una clase con las mismas características, siempre habrá algo que debe identificarlos, por ejemplo la patente. A esto llamaremos identificación y será la o las características que definen de manera unívoca a un objeto.

A continuación podemos ver de manera gráfica la representación de tres objetos distintos que pertenecen a la misma clase Vehículo.

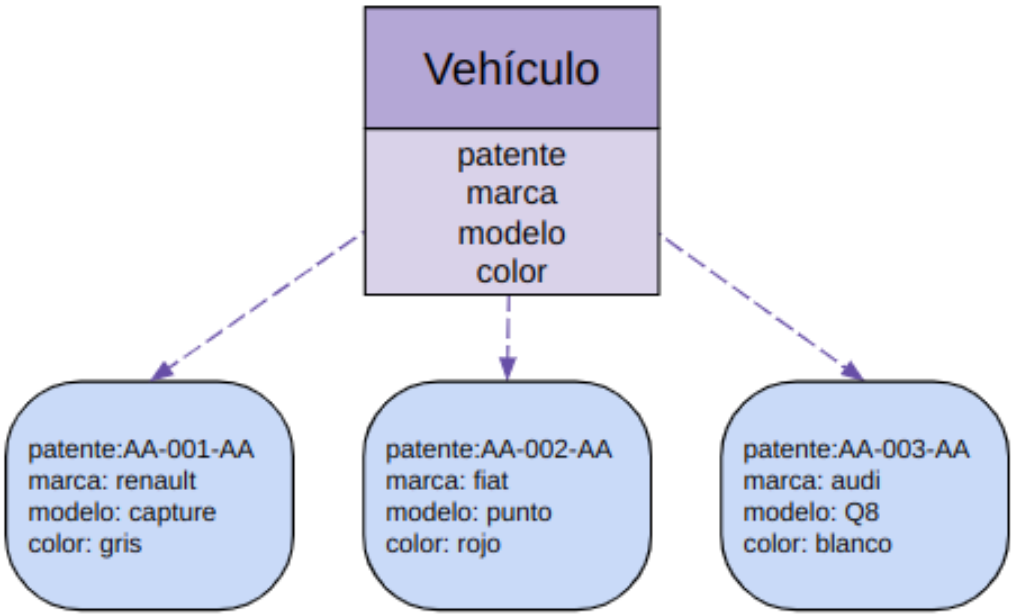



*Fig.32 - clase y objetos.*

Hasta este punto hemos abordado conceptualmente las ideas de clase y objeto. Sin embargo, es importante destacar que sólo hemos enfocado nuestra discusión en las características de estos objetos, sin profundizar en la importancia del comportamiento. La realidad es que la abstracción de objetos de interés no solo se limita a sus atributos, sino que también implica comprender y modelar su comportamiento. Por lo tanto, explorar cómo los objetos interactúan y se comportan en un sistema es un aspecto fundamental que desarrollaremos con mayor profundidad.

# Clase

Una clase es una plantilla o modelo que describe las propiedades y comportamientos de un conjunto de objetos relacionados.

Es un tipo de dato que define un conjunto de atributos y métodos que caracterizan a los objetos creados a partir de ella.

Por el momento cuando hablemos de métodos pensemos en funciones.

En nuestro ejemplo la clase "Vehículo" describe sus atributos y será el medio por el cual crearemos los objetos necesarios.

Veamos algo de código para sustentar estas ideas. La clase Vehículo tendrá la siguiente definición:

```
public class Vehiculo {

   //
   private String patente;
   //
   private String marca;
   //
   private String modelo;
   //
   private String color;

   // Constructor
   public Vehiculo(){
    //…
   }

}
```

La clase “Vehiculo” tiene definidos los atributos patente, marca, modelo y color.

Luego hablaremos de la palabra reservada “private” y de otras relacionadas, por ahora pensemos que sólo los objetos de la clase “Vehiculo” podrán tener acceso a estos atributos.