

# Introducción a la Orientación a Objetos

Jose Jesus García Rueda

(Adaptado de "Object-Oriented Information Systems", de David Taylor)

## Una nueva aproximación a la construcción de software

- La tecnología de objetos se basa en la reutilización de software.
- Tres son sus mecanismos clave:
  - Objetos.
  - Mensajes.
  - Clases.

## Construir el software como el hardware

- El objetivo es obtener el mismo éxito que el mundo del desarrollo hardware.
- Los ordenadores están contruidos en capas de elementos discretos interconectados.
- El software orientado a objetos también se construirá por conexión de elementos discretos en capas.

## Beneficios de la tecnología de objetos

1. Permite desarrollar software en mucho menos tiempo y con menos coste.
2. Se consigue aumentar la calidad de los sistemas.
3. El software orientado a objetos es más fácil de modificar y mantener.
4. La tecnología de objetos facilita la adaptación al entorno y el cambio.

## Los sistemas complejos en la naturaleza

- En la naturaleza encontramos sistemas extremadamente complejos que... funcionan.
- ¿Podremos conseguir esto mismo con los sistemas artificiales si imitamos los mecanismos de la naturaleza?
- Tres son los principios básicos que subyacen en todo sistema vivo.

### 1. Bloques de construcción vivos

- Todo ser vivo está formado por células.

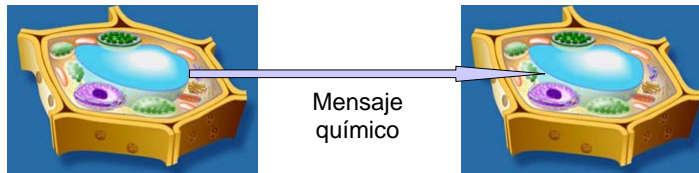


Membrana      Núcleo      Citoplasma

- Dos puntos interesantes:
  - La célula está envuelta por una membrana que la *encapsula*.
  - La célula contiene mecanismos para gestionar tanto la información como el comportamiento.

## 2. Interacción entre células

- Las células interactúan a través de mensajes.

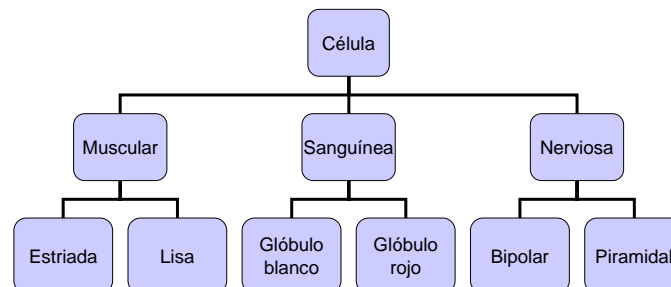


La membrana acepta o rechaza el mensaje  
La célula responde internamente

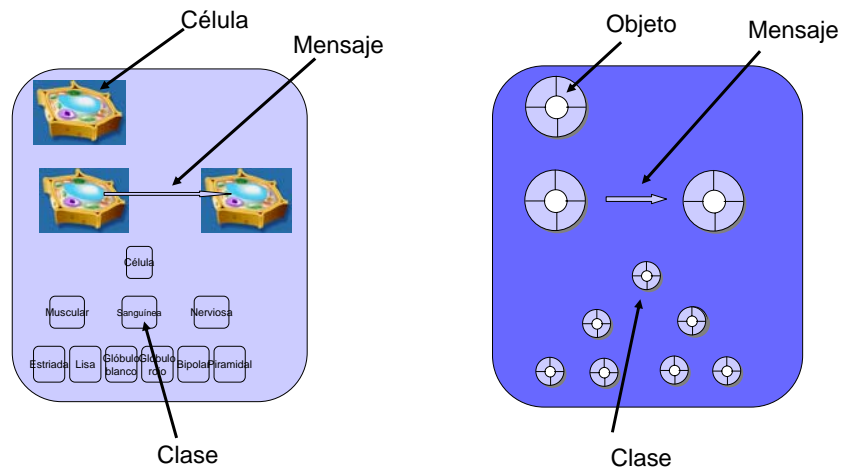
- Este tipo de interacción tiene importantes ventajas:
  - Protege el interior de cada célula de la intrusión de otras.
  - Evita a las células tener que conocer los mecanismos internos de las demás (*ocultación*)

## 3. Especialización de las células

- Cada célula se especializa en una tarea determinada, formando jerarquías.

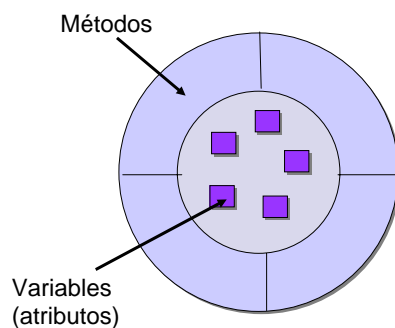


## Las tres claves de la Tecnología de Objetos



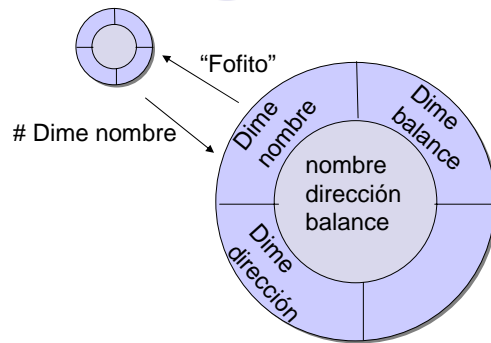
## Objetos

- Los objetos serán la base de todo el software Orientado a Objetos.



- El objeto protege el acceso a sus variables: los objetos sólo pueden accederse a través de sus métodos.

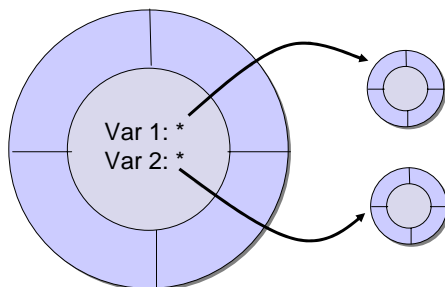
## Un ejemplo de objeto



- Ocultación: si cambio algo dentro del objeto, no tengo que modificar nada fuera.
- Podría incluso eliminar variables...

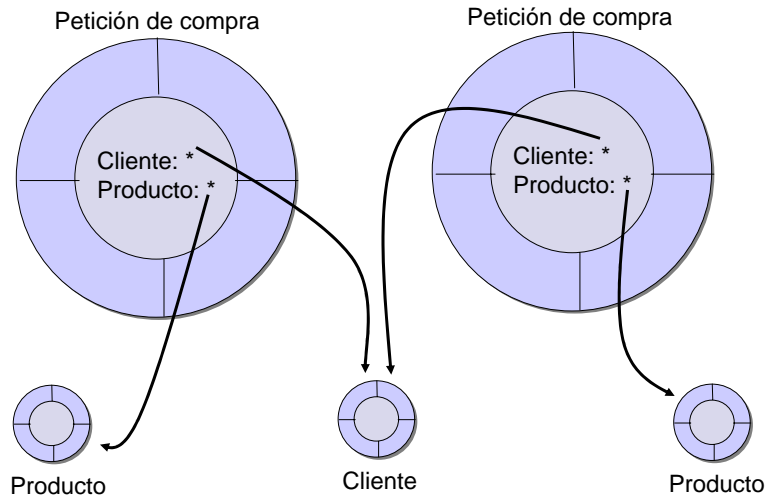
## Construir objetos a partir de objetos

- Un objeto puede contener a otro objetos.
- En la mayor parte de los sistemas, en realidad no los contiene, sino que los *referencia* (apunta)



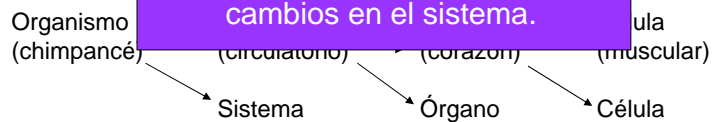
- Ventajas de referenciar:
  - El objeto "contenido" puede cambiar de tamaño y composición sin que eso afecte al contenedor.
  - Los objetos "contenidos" pueden pertenecer a varios objetos a la vez.

## Ejemplo de objetos compuestos



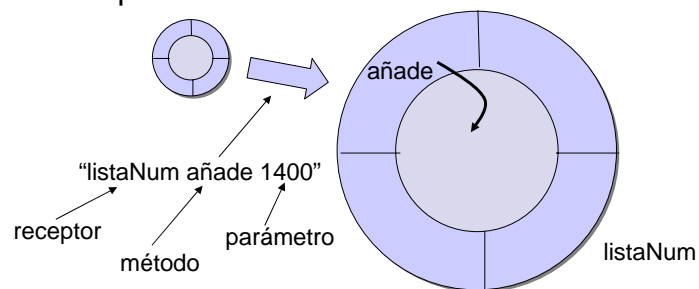
## Modularización por niveles

- Los objetos se diseñan y se implementan en varias partes.
- Esto permite reutilizar los objetos, lo que reduce el tiempo y el costo de desarrollo.
- Ventajas:
  - Cada nivel puede entenderse de forma independiente.
  - Lo mismo con cada componente de cada nivel.
  - Facilita la realización de cambios en el sistema.



## Mensajes

- A través de mensajes los objetos solicitan servicios a otros objetos.
- Un mensaje tiene tres partes: identidad del receptor, método solicitado e información necesaria para el método (*parámetros*)
- Secuencia de actuación: el emisor envía el mensaje, el receptor ejecuta el método apropiado, el receptor devuelve una respuesta.

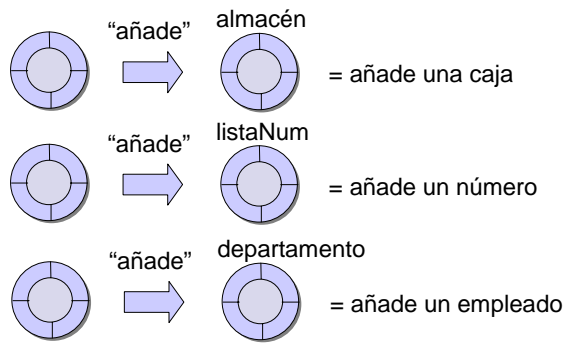


## Ventajas de los mensajes

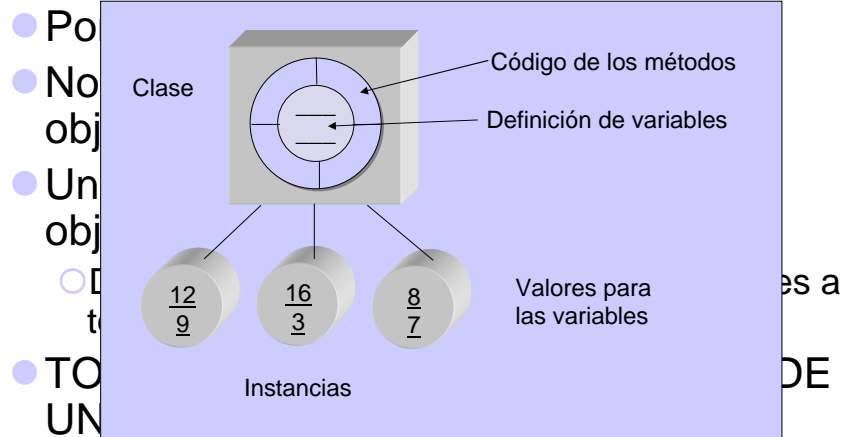
- Los mismos que en la naturaleza:
  - Los objetos no acceden a las "interioridades" de los demás.
  - Los objetos no necesitan conocer las interioridades de los demás.
- Esta encapsulación facilita la modificación de los objetos.

# El poder del polimorfismo

- Se puede usar el mismo nombre para muchos métodos.
- Esto simplifica la escritura de programas, permite que se ejecuten más rápidamente y facilita mucho la modificación.

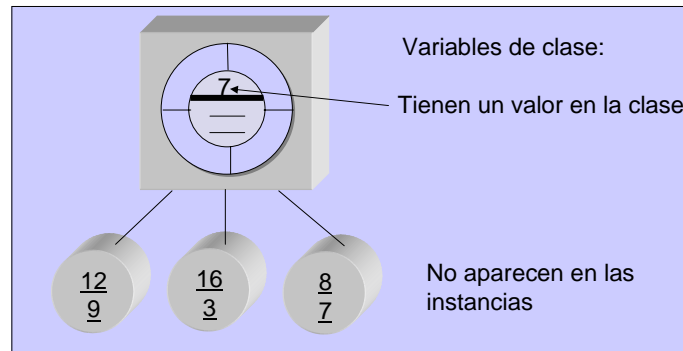


# Clases



## Cuando las clases actúan como “instancias”

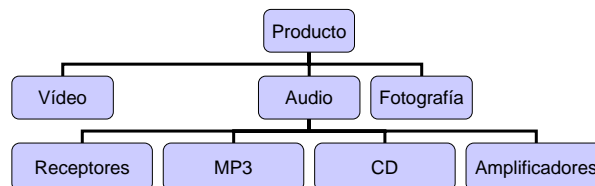
- Las clases pueden contener *variables de clase* (frente a las *variables de instancia*)



- Puede ocurrir lo mismo con los métodos. Ej: constructores y destructores.

## Especialización de clases

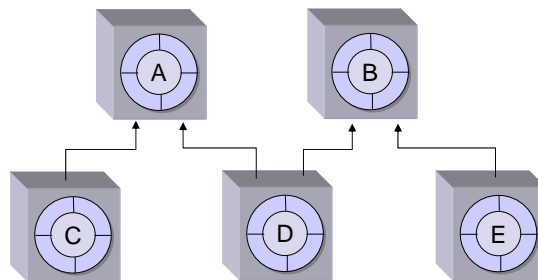
- Cada clase se define como un caso especial de otra, generándose una *jerarquía de clases*.



- Ventaja: las clases pueden compartir propiedades.
  - Se elimina la redundancia.
  - Se facilita la modificación del software.

## Herencia múltiple

- Cuando cada clase tiene una única superclase de la que hereda, tenemos *herencia sencilla*.
- Si se permite que las clases hereden de tantas superclases como se desee, tenemos *herencia múltiple*.



## La naturalidad de las jerarquías de clases

- Las jerarquías de clases reflejan nuestra forma natural de entender:
  - Generalización.
  - Discriminación (especialización)
- Podemos trasladar fácilmente nuestro conocimiento del mundo real a un sistema orientado a objetos.

## Conclusiones



- La Orientación a Objetos es el paradigma actual para la creación de software.
- Se basa en la descomposición del programa en elementos discretos y “blindados” (objetos), que colaboran entre sí enviándose mensajes.
- Los objetos son instancias (ejemplos) de una clase (plantilla)
- La Orientación a Objetos posee una serie de propiedades que aportan importantes ventajas.