Universidad Tecnológica Nacional
FACULTAD REGIONAL CORDOBA

PARADIGMAS DE PROGRAMACION

Unidad III

Paradigma de Programación con Orientación a Objetos

Introducción

Objetivos de la unidad.

- Que el alumno comprenda los mecanismos fundamentales que dan origen a este paradigma.
- Que el alumno utilice para la resolución de problemas un lenguaje con OO puro en su concepción y representación.
- Que el alumno aprenda el lenguaje Smalltalk.

CONTENIDOS ABORDADOS

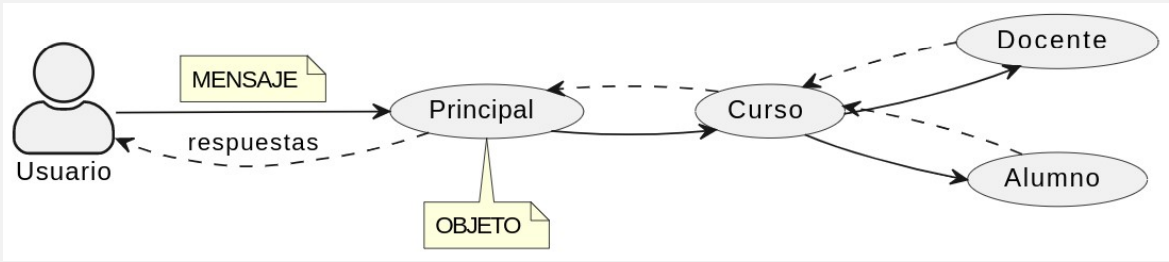
- Paradigma orientado a objetos:
    - Historia.
    - Conceptos.
    - Características.
    - Objeto.
    - Colaboradores.

Historia

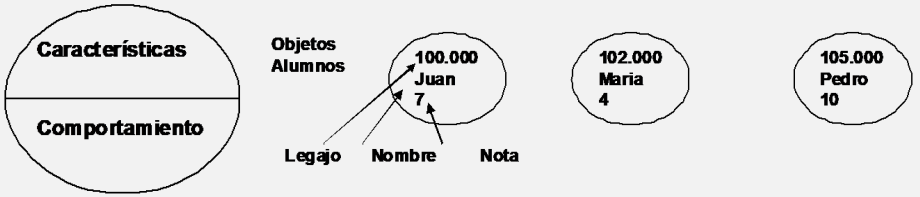
- 1967 : Simula ´67 Primer lenguaje orientado a objetos.
- 1972 : SMALLTAK Adele Goldberg, Simula, Alan Kay.
- 1980 : Objective-C Brad Cox.
- 1984: SMALLTALK-V PC-IBM.
- 1985: EIFFEL Bertrand Meyer.
- 1986 - 1990: C++ Bjarne Stroustrup.
- En la década de los años 1990 se popularizó el uso de la programación orientada a objetos o POO, y es usada hasta la actualidad.

Paradigma Orientado a Objetos

- Los programas se organizan como colecciones de objetos que colaboran entre sí enviándose mensajes.
- Cada objeto tiene un rol específico en el programa y todos pueden comunicarse entre sí de formas predefinidas.



- Un OBJETO es una entidad que posee características (atributos) y comportamientos (métodos).



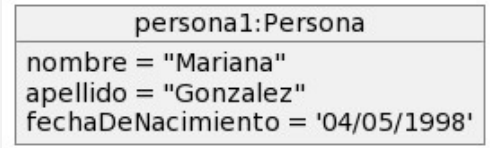
Paradigma Orientado a Objetos

CLASE:

- Es un molde o plantilla que define las características o atributos (variables) y comportamientos o métodos (funciones) comunes de un conjunto de objetos relacionados.
- Describe cómo deben ser los objetos que se crean a partir de ella.

OBJETO:

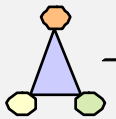
- Es una instancia o ejemplar de una clase.
- Es una entidad concreta que se crea a partir de una clase y tiene sus propios atributos y comportamientos.
- Posee las características generales de la misma, pero con valores concretos.
- Es una variable compuesta por variables y funciones, definida por el usuario.



Características

Abstracción

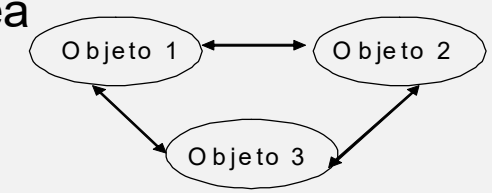
- Denota las características esenciales de un objeto, donde se capturan sus comportamientos.
- El proceso de abstracción permite seleccionar las características relevantes dentro de un conjunto e identificar comportamientos comunes para definir nuevos tipos de entidades en el mundo real.

Persona	Características	Características Esenciales según el ámbiente o Sistema	
	dni, nombre (identificación) teléfono, domicilio, etc. (contacto) altura, peso, etc. (contextura física) Estudios secundario (formación)	**Persona en la universidad** dni, nombre teléfono, domicilio, etc. Estudios secundario, etc.	**Persona en una agencia de Modelos** dni, nombre teléfono, domicilio, etc. altura, peso, etc.

Características

Modularidad

- Permite subdividir una aplicación en partes más pequeñas, cada una de las cuales debe ser tan independiente como sea posible. Se pueden compilar por separado y tienen conexiones con otros módulos.

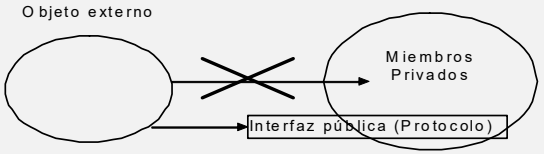


Encapsulamiento

- Significa reunir a todos los elementos que pueden considerarse pertenecientes a una misma entidad, al mismo nivel de abstracción.

Principio de ocultación

- Cada objeto está aislado del exterior y expone una *interfaz* (protocolo) a otros objetos, que específica cómo pueden interactuar con los objetos de la clase.
- El aislamiento protege las propiedades de un objeto contra su modificación por quien no tenga derecho a acceder a ellas.



Características

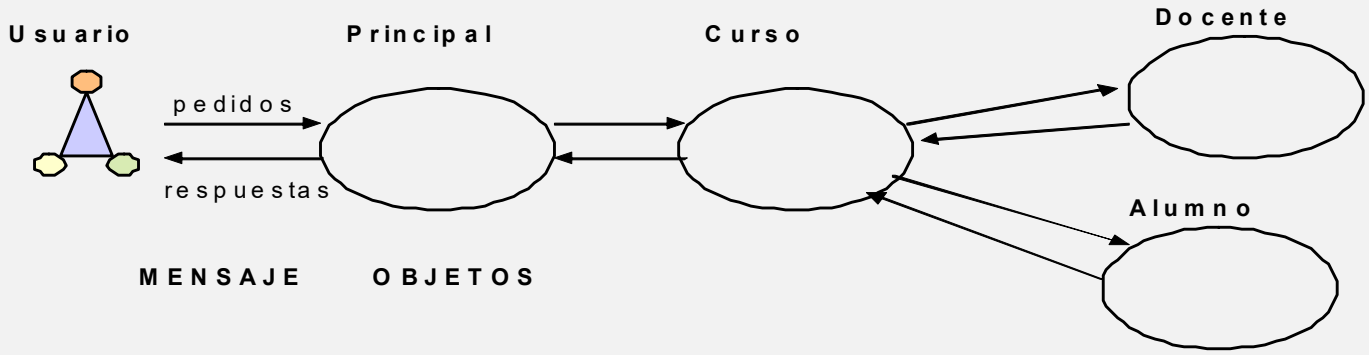
Cohesión

- Mide las responsabilidades asignadas a cada objeto.

Acoplamiento

- Mide las relaciones entre los objetos.

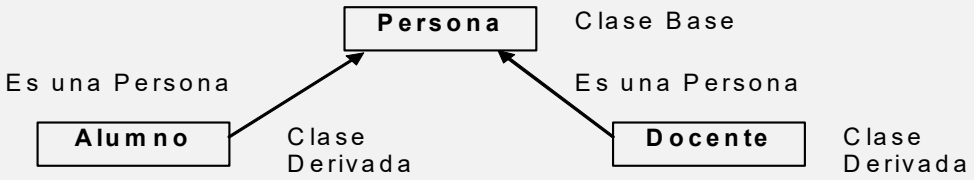
- En el Modelado Orientado a Objetos, lo que se busca es tener
  - ALTA cohesión (que cada objeto se responsabilice por una sola cosa) y
  - BAJO Acoplamiento (Poca o nula interdependencia).



Características

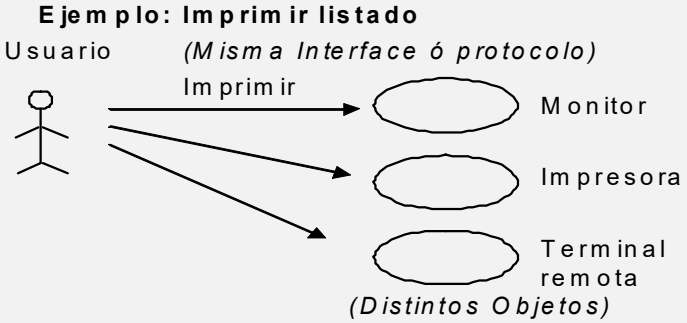
Herencia

- La herencia permite la definición de una clase a partir de la definición de otra ya existente.
- Los objetos heredan las propiedades y el comportamiento de todas las clases a las que pertenecen, pueden compartir (y extender) su comportamiento sin tener que volver a implementarlo.



Polimorfismo

- Capacidad de que diferentes objetos reaccionen de distintas formas a un mismo mensaje.
- Comportamientos diferentes, asociados a objetos distintos, pueden compartir el mismo nombre (interfaz).



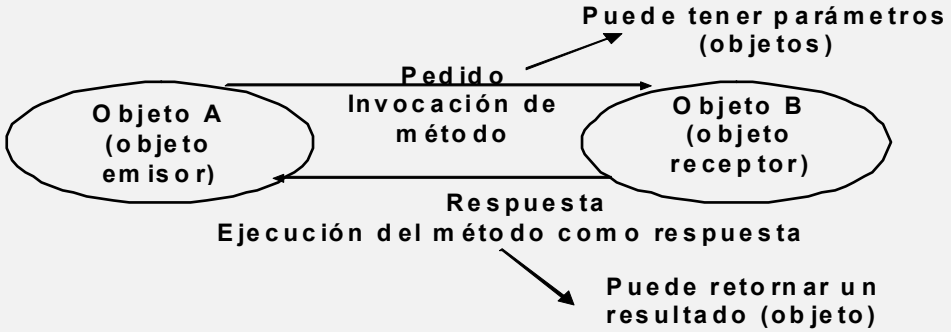
Características

Recolección de basura

- La recolección de basura o *garbage collector* es la técnica por la cual el entorno de objetos se encarga de destruir automáticamente, y por tanto desvincular la memoria asociada, los objetos que hayan quedado sin ninguna referencia a ellos.
- Significa que el programador no debe preocuparse por la asignación o liberación de memoria, ya que el entorno la asignará al crear un nuevo objeto y la liberará cuando nadie lo esté usando.

Objeto

- Es una abstracción, representación computacional de una entidad de la realidad, tiene propiedades particulares (atributos) y formas de operar con ellas (métodos).
- Un objeto es una unidad de software de la cual lo que importa es: qué le puedo preguntar y/o pedir, y a qué otros objetos conoce.
- Los objetos responden a los pedidos interactuando con los otros objetos que conoce, y así se va armando una aplicación.



Los Objetos en POO

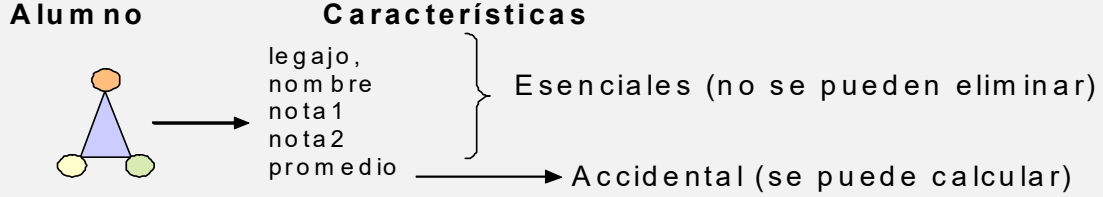
Un objeto dentro de Paradigma Orientado a Objetos debe poseer:

- Abstracción
- Identidad
- Comportamiento
- Capacidad de inspección

Los Objetos en POO

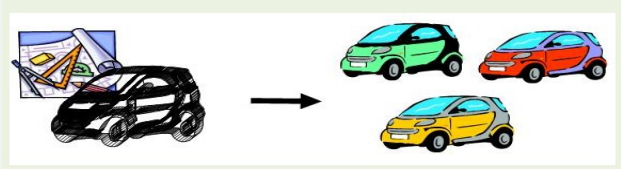
Abstracción

- Un Objeto es una abstracción, del cuál se identifican dos características:
  - **Esenciales**: aquellas que hacen que el ente sea lo que es, que no pueden cambiar, que si falta algo ya no es lo que era.
  - **Accidentales**: Un ente puede poseerlos, pero si no los tuviera o se lo cambiara, no dejaría de ser lo que es. (color, tamaño, etc.).



Identidad

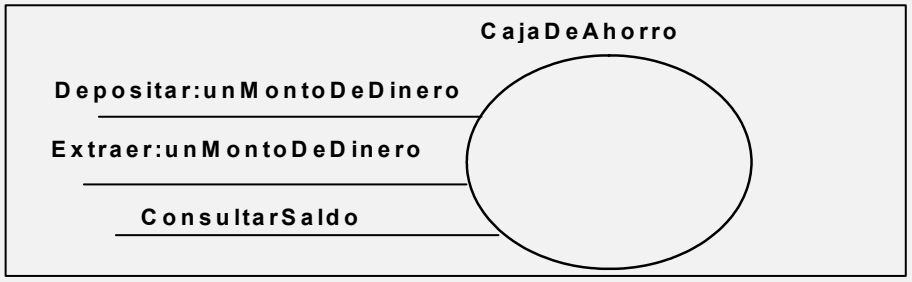
- La identidad es la propiedad que permite a un objeto diferenciarse de otros.
- Generalmente esta propiedad es tal, que da nombre al objeto.
- La identidad de los objetos sirve para comparar si dos objetos son iguales o no.



Los Objetos en POO

Comportamiento

- Un objeto también queda representado por su comportamiento, el cual estará definido por el conjunto de mensajes que el objeto pueda responder (para que sirve, como utilizarlo). Este conjunto de mensajes que el objeto puede responder se lo denomina **Protocolo**.

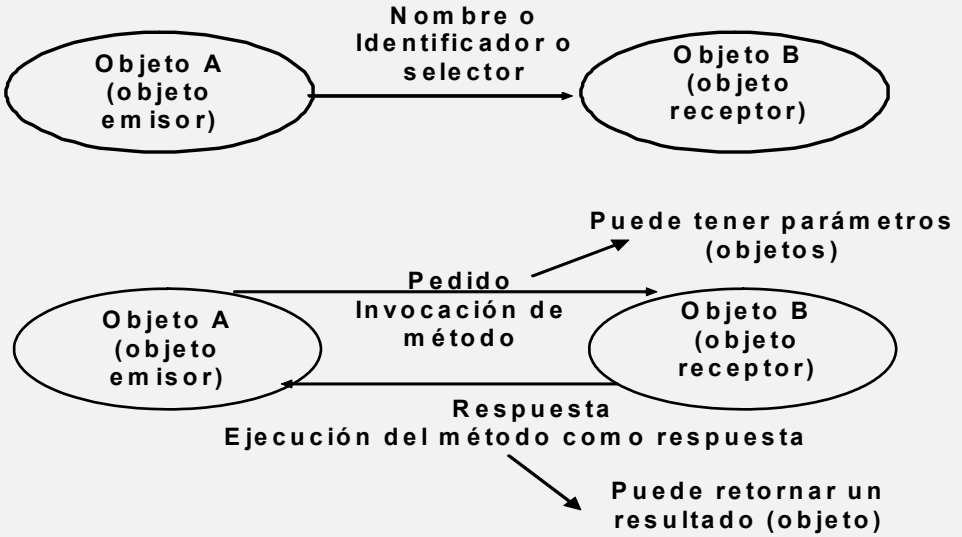


Capacidad de inspección

- Poder ver en todo momento quiénes son sus colaboradores, y en un determinado instante eventualmente cambiárselo.

Colaboraciones: Mensaje y Métodos

- La colaboración se produce cuando, los objetos se envían mensajes.
- **Un mensaje:** es cada una de las formas posibles de interactuar con un objeto.
- Cada objeto entiende un conjunto acotado de mensajes (Protocolo).
- En cada acción de envío de mensaje:
  - hay un emisor.
  - hay un receptor.
  - hay un nombre, que identifica el mensaje que estoy enviando entre todos los que entiende el receptor.

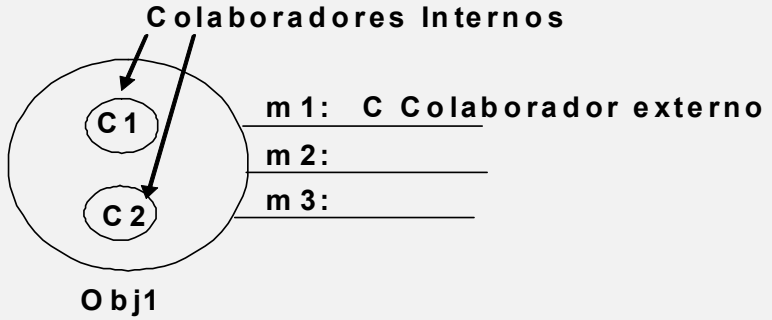


Colaboraciones: Mensaje y Métodos

Método:

- Sección de código que se evalúa cuando un objeto recibe un mensaje.
- Un método tiene un nombre, que es el del mensaje correspondiente; y un cuerpo, que es el código que se evalúa.
- Para poder especificar un **método** debo identificar los colaboradores.
- Tipos :
  - Habituales (internos)
  - Eventuales (externos): se define especialmente para alguno de los mensajes.

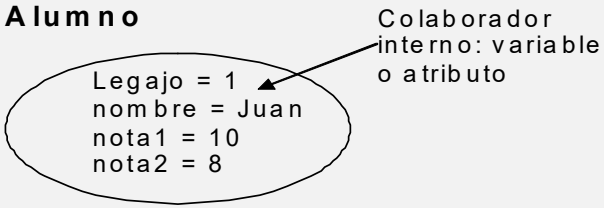
COLABORACION



Colaboraciones

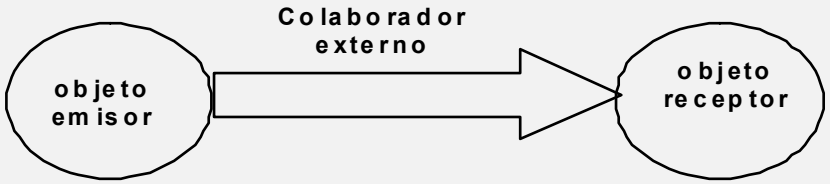
Internos

- El **conjunto de variables que va a contener un objeto** se llama Colaborador Interno o estado interno.
- El **estado interno** de un objeto: es el conjunto de variables que contiene, y se define al programar el Objeto.



Externos

- La colaboración **es un contrato entre emisor y receptor**.
- El emisor debe escribir bien el mensaje y enviar un colaborador externo que sepa hacer cosas que requiere el receptor. Por su lado el receptor debe devolver un objeto que sepa hacer cosas que requiere el emisor.



Colaboraciones externos

- El receptor del mensaje **no tiene idea** de quien le envía el mensaje. Está totalmente desacoplado; los objetos están preparados para responder a cualquiera que le envíe le mensaje.
- El emisor, asume que existe un objeto con capacidad de responder al mensaje que envío.
- El único error que podría ocurrir es que un objeto reciba un mensaje que no entiende:
  - Mensaje equivocado al objeto correcto.
  - Mensaje correcto al objeto equivocado.