PROGRAMACIÓN ORIENTADA A OBJETOS

Dr. Noé Alejandro Castro Sánchez

Introducción

- Nueva filosofía para resolución de problemas:
  - Descomposición de la realidad en objetos.
- Objetos: representación de entidades o conceptos. Poseen:
  - propiedades o características (atributos, datos)
  - comportamiento o acciones (interfaz)
- La forma de comunicarse o hacer peticiones (enviar mensajes) a un objeto es a través de su *interfaz*
  - Tareas que el objeto puede llevar a cabo,
  - Cada tarea representa una función o método.

Ejemplo (mundo real)

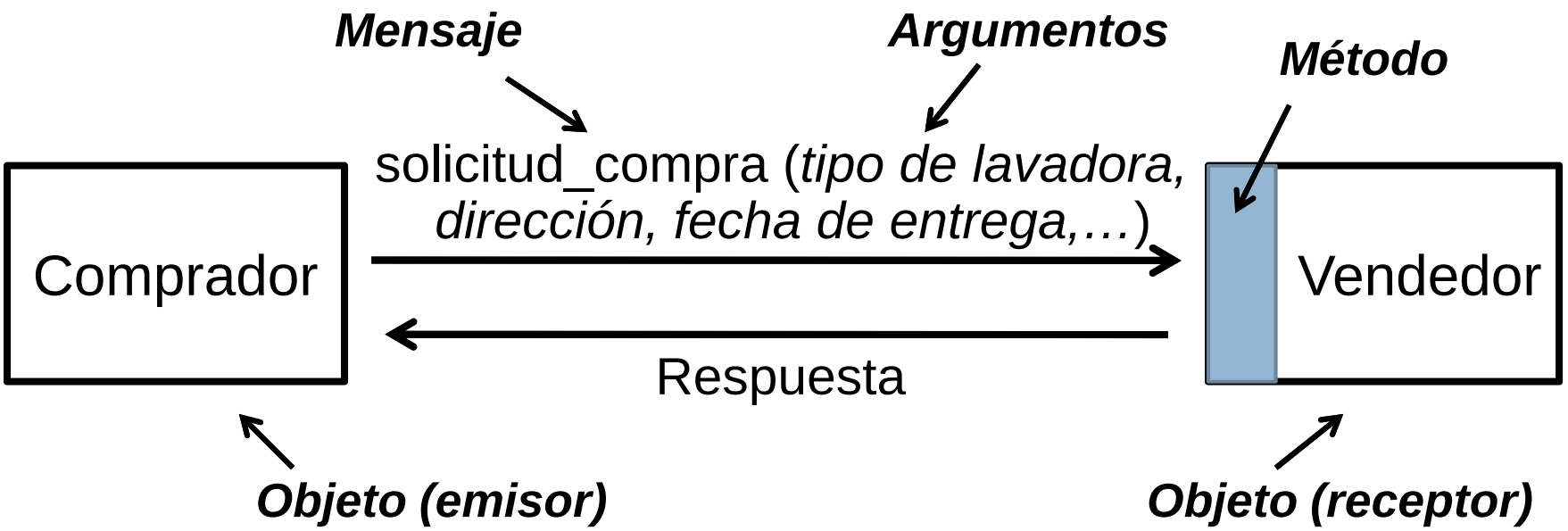
- Solicitud de compra de lavadora:
  - Tipo de lavadora
  - Lugar y fecha de entrega

Petición

Comprador

- Respuesta a solicitud:
  - Determinar existencia
  - Fecha de entrega oportuna

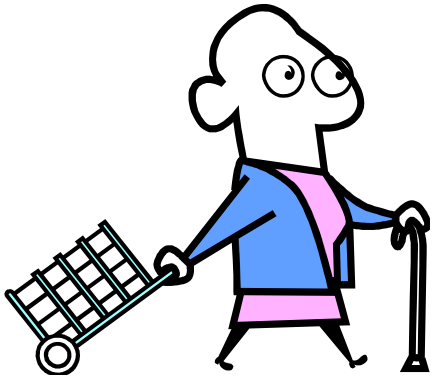
Implementando en POO



- Mensaje válido: corresponde a un método que el objeto receptor tiene implementado.
  - Solicitud de descuento, ¿será un mensaje válido?

Identificando objetos

- Información común en ambos:
  - Atributos: Nombre, dirección, presupuesto (los valores de los atributos representan el estado de un objeto).
  - Método: comprar (envía solicitud de compra al vendedor).



Modelando el ejemplo

- Estructurar la información de uno de los clientes como *objeto*.

```
Ana como Objeto
    Atributos:
        Nombre: “Ana”
        Dirección: “Querétaro Lab. 2”
        Presupuesto: 10,000
    Métodos:
        comprar()
```

Identificando clases

- Clase: plantilla que define atributos y métodos comunes a todos los objetos de cierto tipo (se comparte una misma estructura entre objetos similares):
  - Clase *compradores*.
- Un objeto es una *instancia* de una clase. Los atributos adquieren sus propios valores en cada objeto.
  - Cada comprador en particular es una instancia de la clase *compradores* y tiene sus propios valores.

Resumen mensajes y métodos

- Mensajes: llamadas a métodos desde un objeto emisor a un objeto receptor.
- La llamada a un método puede ir o no acompañada de argumentos.
- El conjunto de métodos de un objeto define su comportamiento.
- Un mensaje es válido si el receptor tiene un método que corresponde al nombrado en el mensaje y a los argumentos correspondientes.

Ejercicio 1

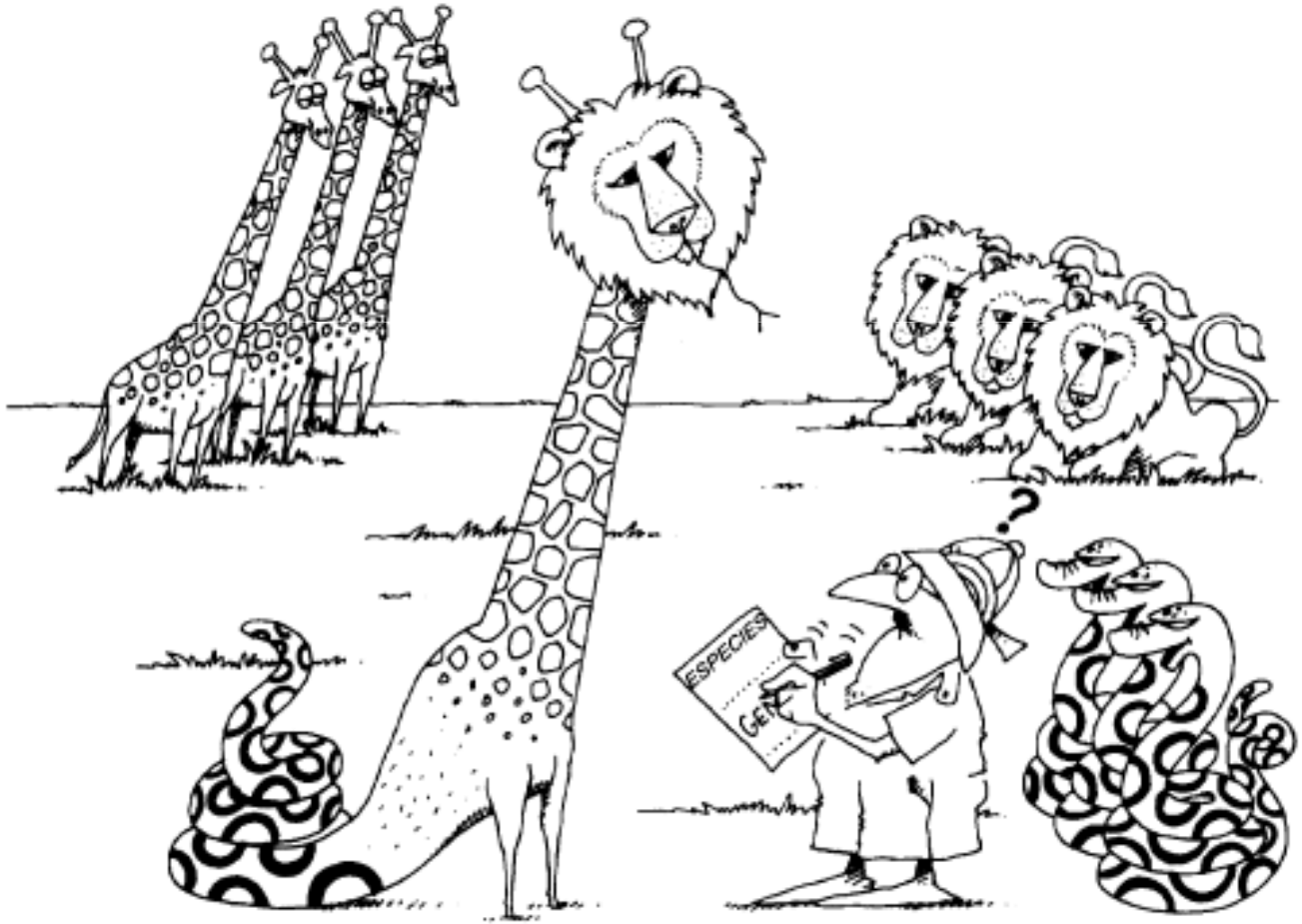
- Contador con las siguientes características:
  - Dos botones y pantalla para 5 dígitos.
  - Botón 1: incrementa el valor del contador en uno.
  - Botón 2: reinicia (hace a cero) el valor del contador.
- Crear la clase correspondiente al contador
  - identificar atributos y métodos
- Crear dos objetos de la clase

¿Por qué *clasificar*?

Importancia de la clasificación

- Agrupar cosas que tienen una estructura común y/o exhiben un comportamiento similar.
- La identificación de clases y objetos es la parte más difícil del diseño OO.
- Proporciona una guía para tomar decisiones sobre modularización.

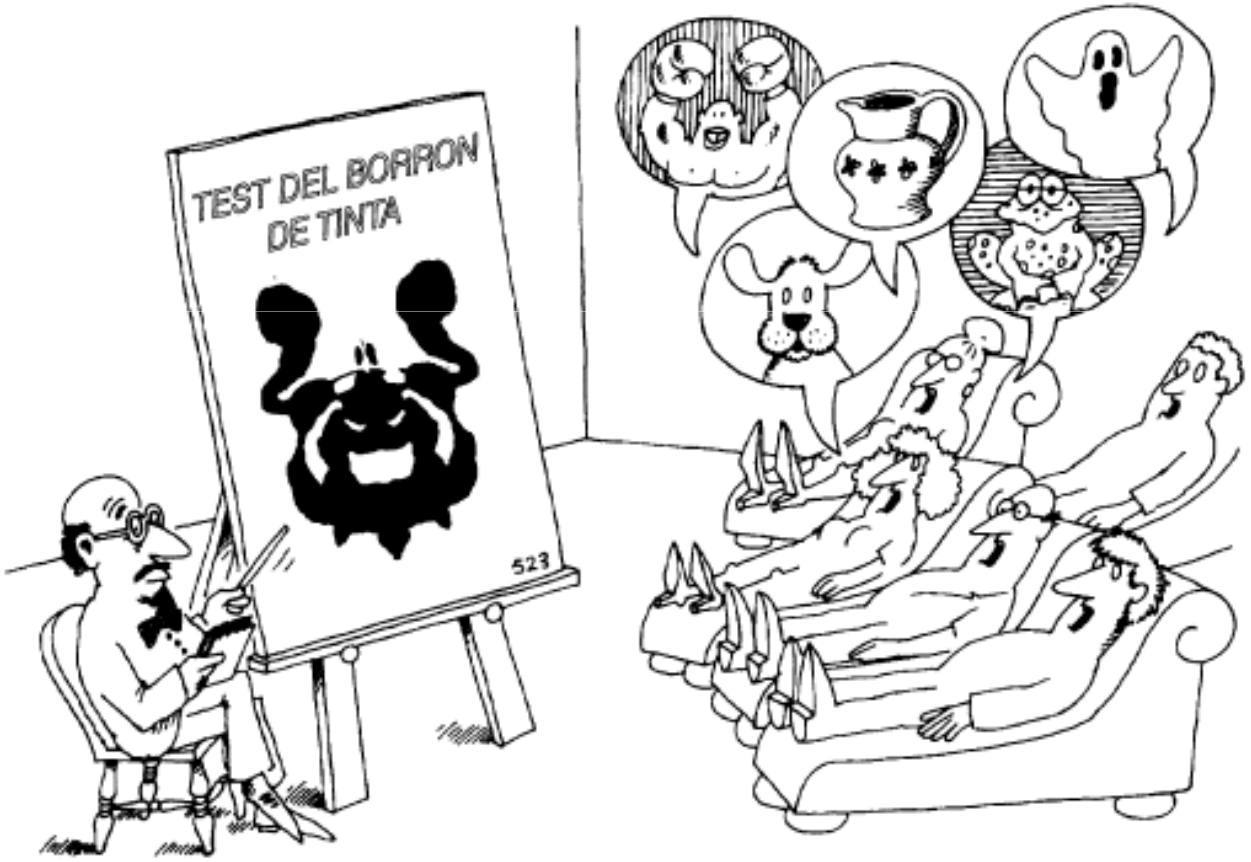
Importancia de la clasificación (II)



Importancia de la clasificación (III)

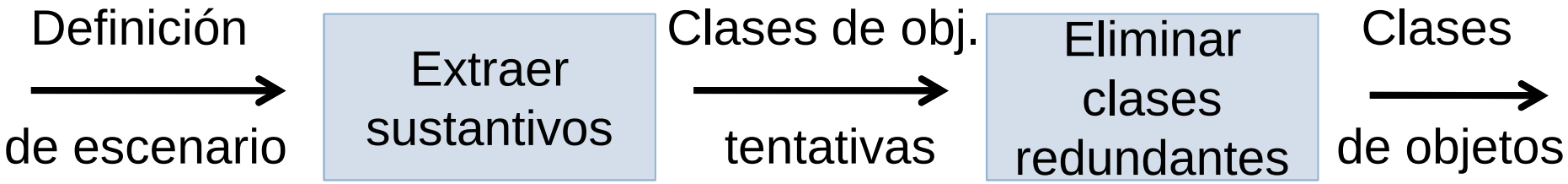
- Llega a ser difícil porque:
  - En teoría, la interfaz de un objeto debe ser clara y debe estar bien delimitada.
  - A veces sus fronteras son difusas.
  - La clasificación depende de la perspectiva del observador.
- No hay clasificaciones perfectas, pero sí hay unas mejores que otras.

Importancia de la clasificación (IV)



Identificación de clases y objetos

- Enumerar candidatos a clases
  - Escribir el problema (escenario del mundo) y subrayar los sustantivos.
- Eliminar clases redundantes, irrelevantes y vagas.
- Identificar los atributos.



Lenguaje UML

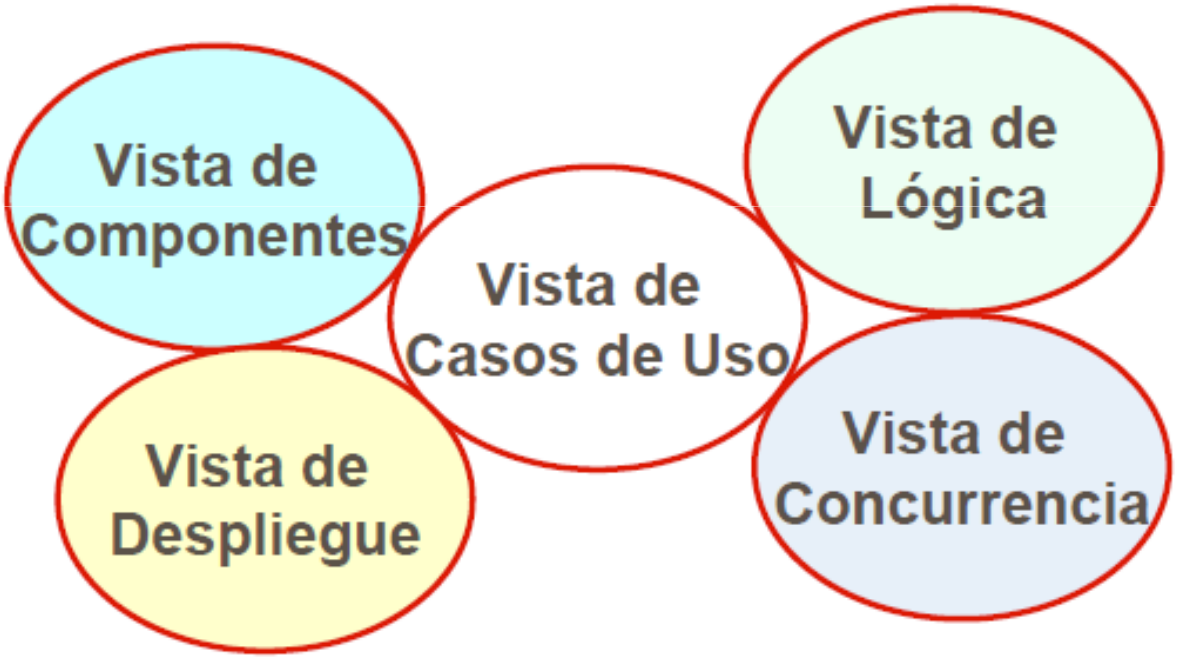
Modelado de sistemas

- Lenguaje UML: (Unified Modeling Language) estándar que permite el análisis y diseño para el desarrollo de aplicaciones OO.
- Consta de **diagramas** para visualizar, especificar, construir y documentar componentes de un sistema.
- Los diagramas describen *vistas*.
  - Vista: representa una proyección del sistema, mostrando aspectos particulares del mismo.

Pros vs contras

- Objetivos:
  - Comunicar la estructura de un sistema complejo
  - Especificar el comportamiento deseado del sistema
  - Comprender mejor lo que estamos construyendo
  - Descubrir oportunidad de simplificación y reutilización
- Inconvenientes:
  - Excesivamente complejo: “el 80% de los problemas puede modelarse usando 20% de UML”.

Vistas de UML



Diagramas de UML

- Estáticos
  - Diagrama de Clases
  - Diagrama de Objetos
  - Diagrama de Casos de uso
  - Diagrama de Componente
  - Diagrama de Implantación
- Dinámicos
  - Diagrama de Interacción (secuencia y colaboración)
  - Diagrama de Estados
  - Diagrama de Actividad

Vista lógica

- Funcionalidad del sistema (estructura y comportamiento).
- Descripción de estructura:
  - Diagramas de clases y objetos
- Descripción de comportamiento:
  - Diagramas de interacción: diagramas de secuencia y de colaboración.

Diagrama de clases

- Describe la estructura de las clases y las relaciones entre ellas.

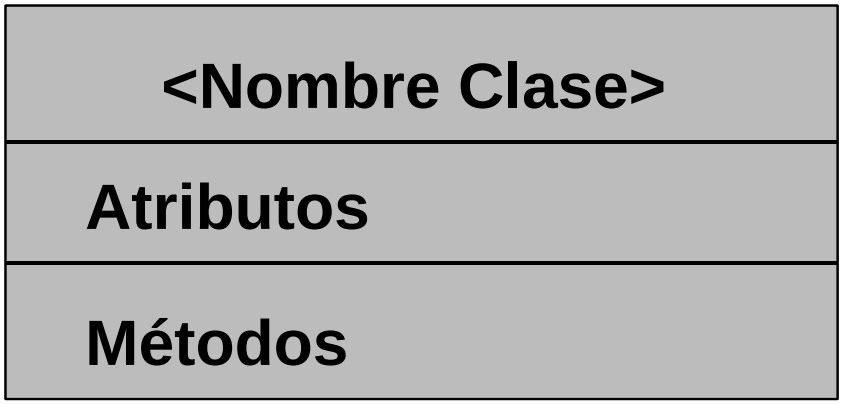


Diagrama de objetos

- Caso especial del diagrama de clases.
  - Cubren la vista de diseño pero desde la perspectiva de casos reales o prototípicos.
  - Especie de fotograma de un instante en tiempo de ejecución.

<Nombre_objeto :Clase>

< :Clase>

Diagramas de interacción

- Modelan aspectos dinámicos del sistema.
- Muestran interacciones: conjunto de objetos, sus relaciones y los mensajes que se envían entre ellos.
- Contiene: objetos, enlaces y mensajes.
- Diagrama de secuencia: ordenación temporal de los mensajes.
- Diagrama de colaboración: destaca la organización estructural de los objetos que envían y reciben mensajes

Ejercicio II

- Se modelará la interacción existente entre cajeros automáticos y usuarios:
  - Los cajeros automáticos son dispositivos que aceptan tarjetas de débito, interactúan con usuarios y se comunican con un ordenador central de su respectivo banco para llevar a cabo transacciones (consultar saldo, pagar servicios, retirar dinero, cambiar nip, etc.). El acceso a esta funcionalidad cumple ciertas características de seguridad.