Introducción a UML. Diagramas de clases

Entornos de desarrollo

Diagrama de clases

- Modela los conceptos del dominio de la aplicación.
- Permite visualizar las relaciones entre las clases que involucran el sistema
- Es el diagrama más común en modelos orientados a objetos.
- Elementos de un diagrama de clases:
  - ✓ Clases
  - ✓ Interfaz, clases abstractas, clases parametrizadas
  - ✓ Relaciones de dependencia, generalización y asociación
  - ✓ Colaboraciones
  - ✓ Notas y constraints

Diagramas de clases: Ejemplo

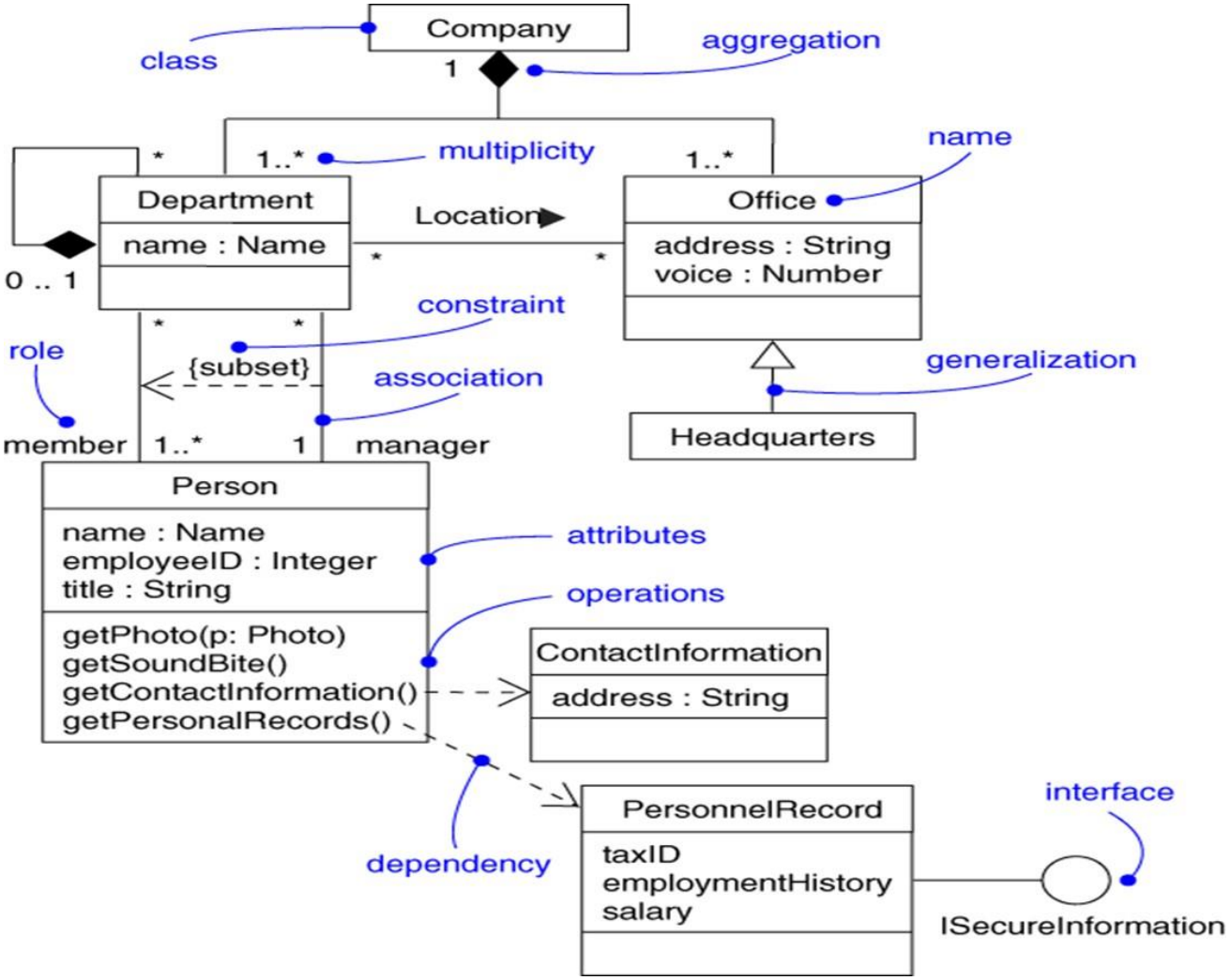


Diagrama de clases: ¿Qué es una clase?

- Es la unidad básica que **encapsula** toda la información de un tipo de objeto (*un objeto es una instancia de una clase*).
- Las clases tienen:
  - **Atributos** son las características individuales que diferencian un objeto de otro y determinan su apariencia, estado u otras cualidades.
  - **Métodos**: son un conjunto de instrucciones que realizan una determinada tarea

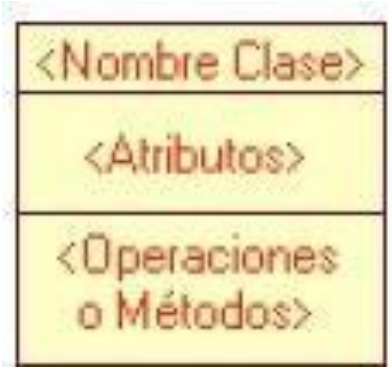


Diagrama de clases: Encapsulación

¿Qué es la encapsulación?

- Ocultamiento de los datos de un objeto de tal forma que solo sean accesibles mediante operaciones definidas por el propio objeto.

  Es el diagrama más común en modelos orientados a objetos.
- La encapsulación presenta una serie de ventajas:
  - Se protegen los datos privados del objeto de lecturas y escrituras no permitidas.
  - Permite una mejor estructuración y manipulación de los datos.
- Los atributos de un objeto no deberían ser manipulables directamente por el resto de los objetos. En caso de querer hacerlos manipulables, se debe implementar los procedimientos Setters y Getters

Diagramas de clases: Visibilidad.

Los atributos describen características de una clase. Existen varios niveles de visibilidad:

- ✓ **Nivel público(public). Símbolo** (+): Las subclases podrán utilizar los atributos y métodos públicos de la clase base y las subclases de estas también los heredarán.
- ✓ **Nivel protegido (protected). Símbolo** (#): Las subclases podrán utilizar los atributos y métodos protegidos de la clase base, pero las subclases de estas no los heredarán.
- ✓ **Nivel privado (private). Símbolo** (-): Las subclases no podrán acceder a los atributos y métodos de la clase base.

Diagramas de clases: Visibilidad.

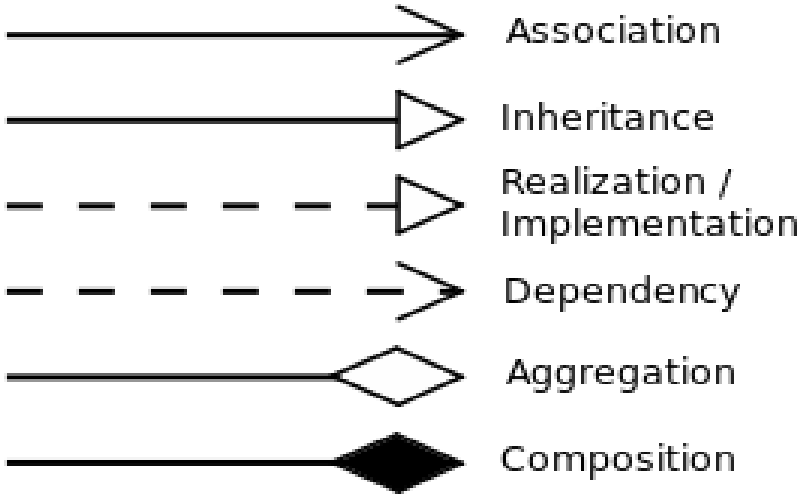
Ejemplo:

- El métodoA() puede acceder al atributoA.
- El métodoB() puede acceder al atributoA.
- El métodoB() puede acceder al métodoA().
- El métodoC() no puede acceder al métodoB().
- El métodoC() puede acceder al métodoA().
- El métodoC() puede acceder al atributoB.
- El métodoC() no puede acceder al atributoA.

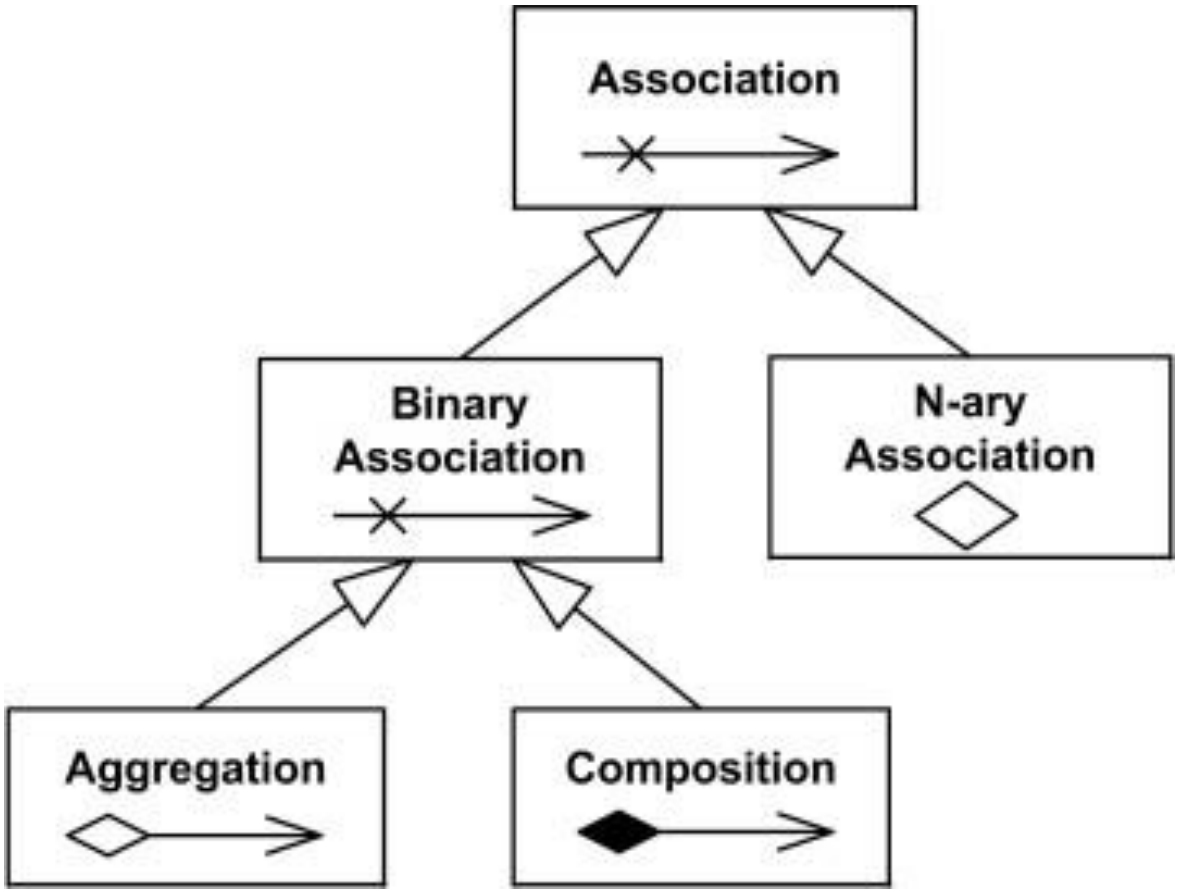
Relaciones entre clases

Las interacciones entre las clases quedan definidas por las relaciones entre ellas.

- Tipo de relaciones:
  - ✓ Asociación
  - ✓ Generalización o herencia
  - ✓ Realización
  - ✓ Dependencia
  - ✓ Agregación
  - ✓ Composición

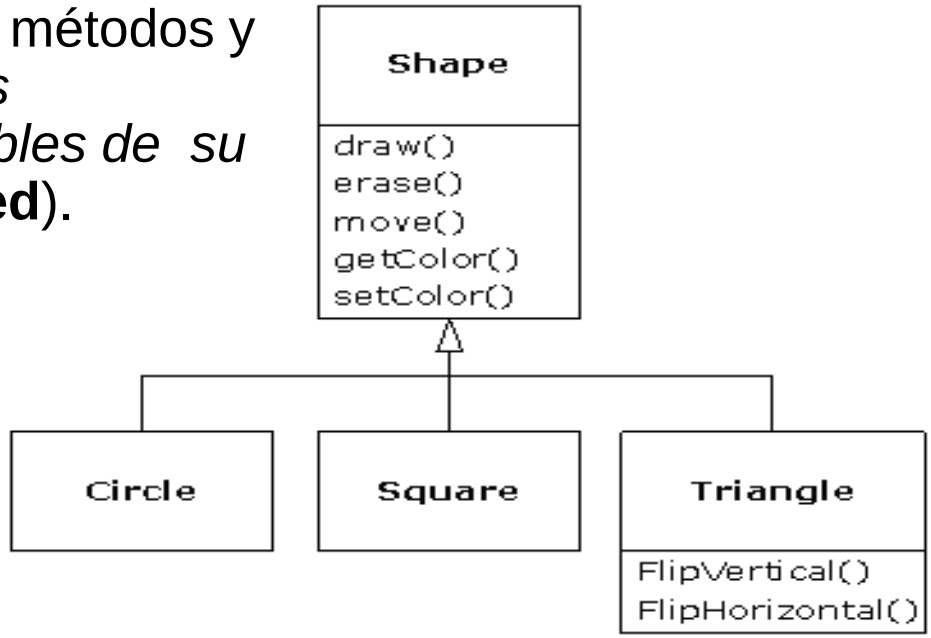


Tipos de asociaciones:



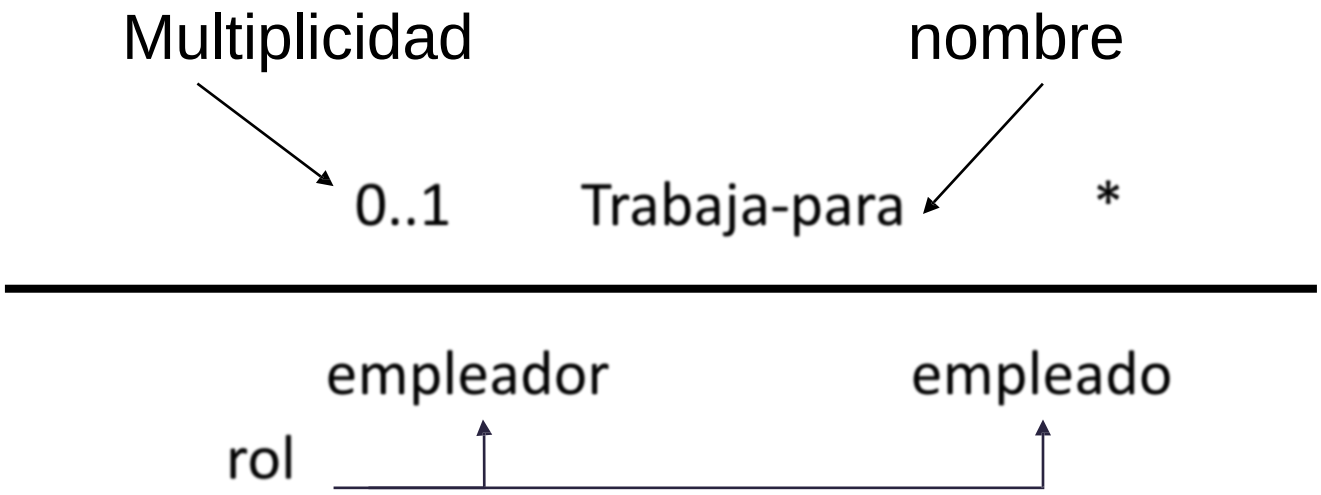
Relaciones entre clases: Generalización o herencia

Indica que una clase (*clase derivada*) **hereda** los métodos y atributos especificados por una clase (**clase base**), por lo cual una clase derivada además de tener sus propios métodos y atributos, *podrá acceder a las características y atributos visibles de su clase base* (**public y protected**).



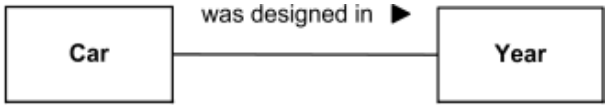
Relaciones entre clases: Asociación

- Una asociación es una relación estructural que especifica que objetos de una clase están conectados a objetos de otra
- Es una relación donde todos los objetos tienen su propio ciclo de vida y **no hay propietario**.
- Multiplicidad:
  - Unaria
  - Binaria
  - N-aria

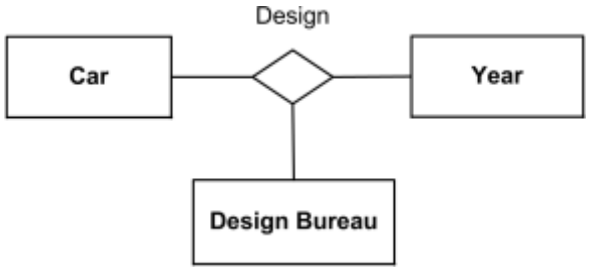


Relaciones entre clases: Asociaciones

- **Rol**
  - ✓ Cuando una clase participa en una asociación, tiene un rol específico que juega en tal relación.
- **Multiplicidad**
  - ✓ Puede ser un rango de valores o un valor explícito:
  - ✓ Exactamente 1 — 1
  - ✓ Cero 0..1 — 0..1
  - ✓ Cero o más — 0..*
  - ✓ Uno o más — 1..*
  - ✓ Subrango — m..n



Relación binaria

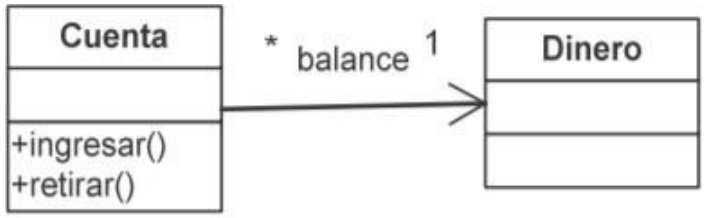


Relación ternaria

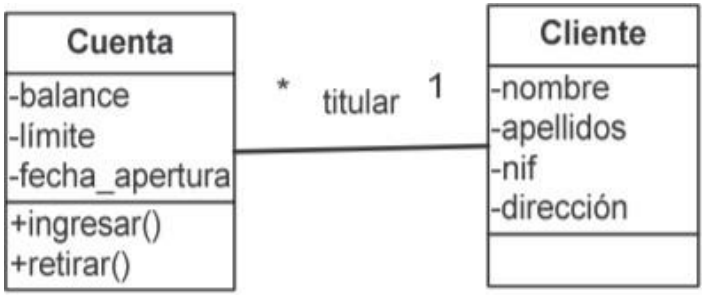
Relaciones entre clases: Asociación

Navegabilidad

- ✓ La navegabilidad es una propiedad del rol. Indica la posibilidad de ir desde el objeto fuente al objeto destino.
- ✓ En un extremo de una asociación se puede indicar la navegabilidad mediante una flecha. Significa que es posible navegar desde el objeto de la clase origen hasta el objeto de la clase destino y por tanto puede llamar a alguna de sus operaciones.



Asociación unidireccional



Asociación bidireccional

Diagramas de Clase: Agregación

- **Agregación** (◇Diamante blanco) es un tipo de asociación que indica que **una clase es parte de otra clase (composición débil)**.
  - ✓ Los componentes pueden ser compartidos por varios compuestos.
  - ✓ La destrucción del compuesto no conlleva la destrucción de los componentes.
- La agregación es una forma especializada de asociación donde todos los objetos **tienen su propio ciclo de vida**, pero **hay propiedad**.

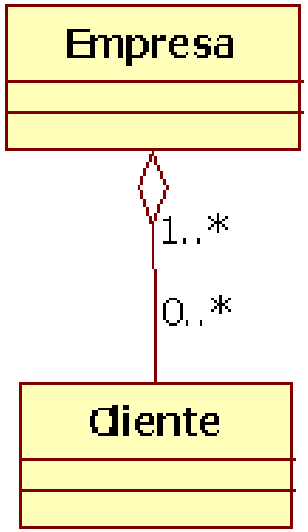
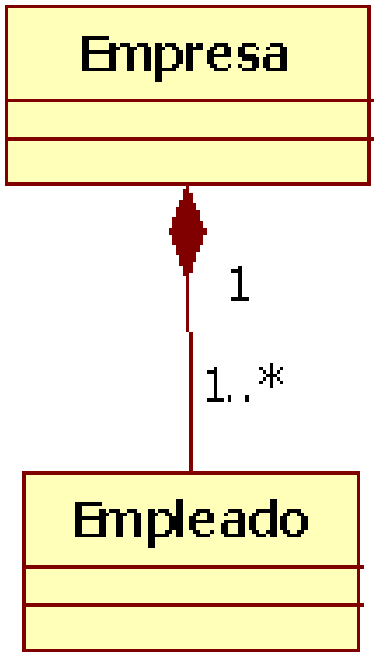


Diagrama de Clases : Composición.

- **Composición** (◆Diamante negro) es una forma fuerte de agregación donde la vida de la clase contenida debe coincidir con la vida de la clase contenedor.
  - ✓ Los componentes constituyen una parte del objeto compuesto. De esta forma, los componentes no pueden ser compartidos por varios objetos compuestos.
  - ✓ El objeto secundario no tiene su ciclo de vida y, si se elimina el objeto principal, también se eliminarán todos los objetos secundarios.



Asociación vs Composición vs Agregacion

- **Asociación:** Varios estudiantes pueden asociarse con un solo maestro y un solo estudiante puede asociarse con varios maestros, pero no hay propiedad entre los objetos y ambos tienen su propio ciclo de vida. Ambos se pueden crear y eliminar de forma independiente.
- **Agregación:** Un solo profesor no puede pertenecer a varios departamentos, pero si eliminamos el departamento, el objeto del profesor no se destruirá. Podemos pensar en ello como una relación \"tiene-a\".
- **Composición:** Una casa puede contener varias habitaciones: no hay vida independiente de la habitación y cualquier habitación no puede pertenecer a dos casas diferentes. Si borramos la casa – la habitación se borrará automáticamente.