

Ingeniería de software II

Diseño orientado a objetos



Diseño orientado a objetos

- ▶ La década de los 90: la era de la programación orientada a objetos (OO).
 - **Necesidad de este paradigma:** los usuarios demandan programas y entornos de trabajo simples y fáciles de usar.
 - Proporciona mejores herramientas para:
 - Obtener un modelo del mundo real cercano a la perspectiva del usuario.
 - Interaccionar fácilmente con un entorno de computación, empleando metáforas familiares.
 - Facilitar la modificación y la extensión de los componentes sin codificar de nuevo desde cero.

Diseño orientado a objetos

- ▶ Existen diferentes modelos:
 - Modelos estructurales: describen la estructura estática del sistema utilizando clases de objetos y sus relaciones
 - Diagrama de clases
 - Modelos dinámicos: describen la estructura dinámica del sistema y muestran la interacción entre los objetos del sistema
 - Diagrama de transición de estado
 - Diagrama de interacción



Diseño orientado a objetos

- ▶ Un sistema orientado a objetos contiene objetos que interactúan, que mantienen su propio estado local y proveen operaciones en ese estado.
- ▶ Las representaciones de su estado son privadas y no pueden accederse desde afuera del objeto.
 - Los objetos están asociados con cosas
 - Se presenta un mapeo entre entidades del mundo real y los objetos en el sistema
 - Los objetos mejoran la comprensión y el mantenimiento del diseño

Focaliza su desarrollo en el concepto
objeto



Diseño OO

- ▶ En la programación orientada a objetos las entidades centrales son **objetos**.
 - Los objetos se comunican entre sí mediante el uso de **mensajes** y el conjunto de objetos que responden a los mismos mensajes se implementan mediante clases.
 - La clase describe e implementa todos los **métodos** que capturan el comportamiento de sus **instancias**.
 - La implementación está totalmente oculta (**encapsulada**) dentro de la **clase**, de modo que puede ser extendida y modificada sin afectar al usuario.
 - Una **clase** es como un módulo
 - Es posible extender y especializar una clase (**herencia**).

Principios básicos

▶ Modularización:

- Módulos fáciles de manejar
- Comprenden las estructuras de datos y las operaciones.

▶ Encapsulado:

- Distingue entre la interfaz de un objeto (qué es lo que hace), de la implementación (cómo lo hace).

▶ Tipos de datos abstractos:

- Agrupa todos los objetos que tienen la misma interfaz y los trata como si fueran del mismo tipo.

▶ Herencia:

- Reutilización.

Principios básicos

► Mensajes:

- Un objeto lleva a cabo sus acciones cuando recibe un mensaje concreto, codificado de una forma simple, estándar e independiente de cómo o dónde está implementado el objeto.

► Polimorfismo:

- Diferentes objetos responden al mismo mensaje.
- El sistema determina en tiempo de ejecución qué código invocar dependiendo del tipo de objeto.

Diagrama de clases

- ▶ Constituyen una *expresión* del modelo OO.
- ▶ Los elementos de un diagrama de clases son:
 - Clases, atributos, operaciones
 - Relaciones, cardinalidad, roles

Cuando se construye un modelo lo primero que se mira es el mundo real, se identifica los objetos esenciales y se representan como clases escribiéndolos en cajas

Diagrama de clases

► Objeto

- Un objeto es un concepto, abstracción o cosa que tiene un cierto significado para una aplicación.
- Un **objeto** es una **instancia** (u ocurrencia) de una *clase*.

Diagrama de clases

► Clase

- Una clase es una descripción de un grupo de **objetos** que tienen:
 - **Propiedades** similares (**atributos**)
 - **Comportamiento** común (operaciones y diagrama de estado) y semántica común.
 - Establece el mismo tipo de **relaciones** con otros componentes del modelo
- Las clases proporcionan un mecanismo para compartir la estructura entre objetos similares.

Seleccionar el esquema de clasificación más apropiado es uno de los aspectos mas importantes el análisis y diseño Orientado a Objetos.

Diagrama de clases

► Atributo

- Un atributo es una propiedad de una clase a la que se le asigna un nombre y que contiene un valor para cada objeto de la clase

► Ejemplo:

- Atributos de Empleado:

- nombre
- Apellido
- Direccion
- Email
- Ciudad
- Cargo/puesto

Para cada objeto de la clase Empleado los valores de los atributos se modificarán.

Juan Perez Urquiza
2538
jperez@gmail.com
operario



Atributos de una clase

- ▶ Ejemplo:
 - Atributos de la clase Perro?
 - Nombre
 - Raza
 - Color
 - Peso
 - Edad
 - Dueño

Para cada objeto de la clase Perro los valores de los atributos se modificarán.

Mika, labrador, chocolate, 90 días,
Mario Fernández



Diagrama de clases

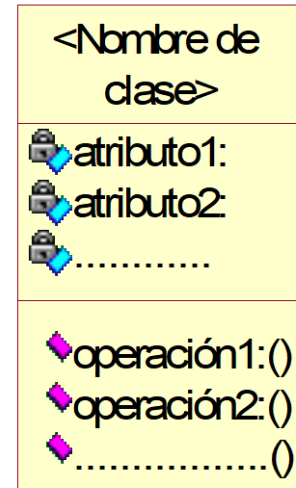
► Identificación de las clases:

- Identificar objetos esenciales
- Refinamiento
 - Las descripciones de Casos de Uso colaboran en la identificación de objetos y operaciones en el sistema.
 - Análisis gramatical de una descripción en lenguaje natural del sistema a ser construido.
 - Objetos y atributos son sustantivos
 - Operaciones y servicios son verbos
 - Usar entidades del dominio de aplicación
 - Utilizar el análisis basado en escenarios
 - identificar objetos, atributos, operaciones.

Clase

Descripciones de un conjunto de objetos que comparten:
atributos, operaciones, **relaciones**
y **semántica**.

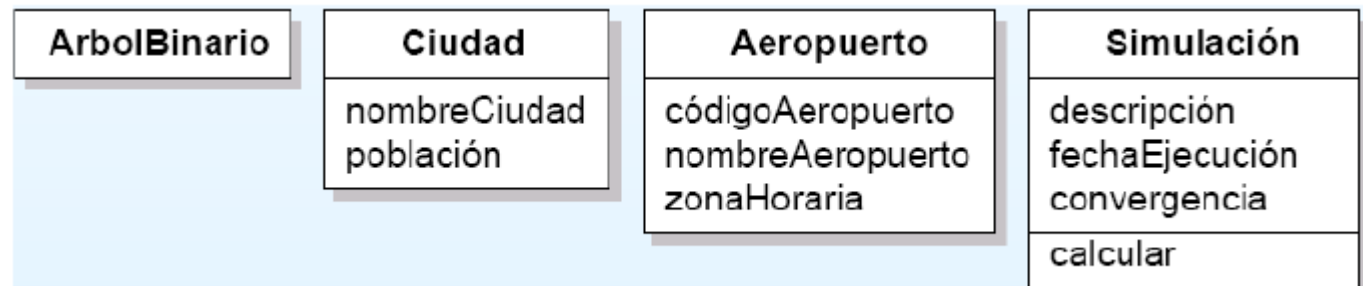
Nombre de la clase
Singular
Inicial mayúscula
Nombre compuesto



- ▶ Debe tener un nombre único
 - Un atributo expresa una propiedad particular de una clase
 - Una operación refleja un servicio del objeto
 - Los atributos y operaciones pueden ser mostrados de manera exhaustiva o no en los compartimentos de clases

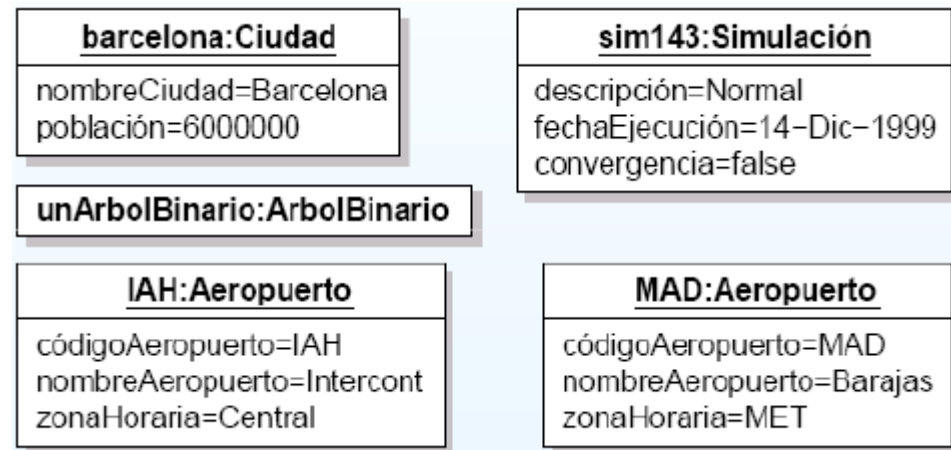
Clase

- ▶ Algunas clases tienen una contrapartida real (persona y empresa por ejemplo).
- ▶ Otras son entidades conceptuales (ecuación algebraica).
- ▶ Las clases y sus relaciones se describen mediante diagramas de clases:



Objetos

- ▶ Los objetos y sus relaciones se describen mediante **diagramas de instancias**.
 - unArbolBinario pertenece a la clase ArbolBinario y no se han especificado los valores de los atributos.
 - El objeto IAH pertenece a la clase Aeropuerto y tiene los valores IAH, Intercont y Central.

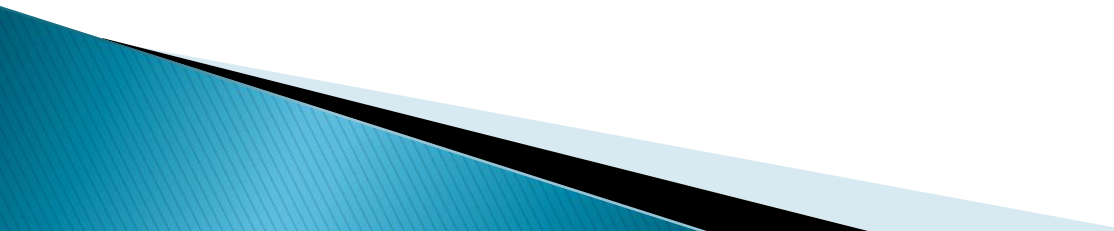


Operaciones y métodos

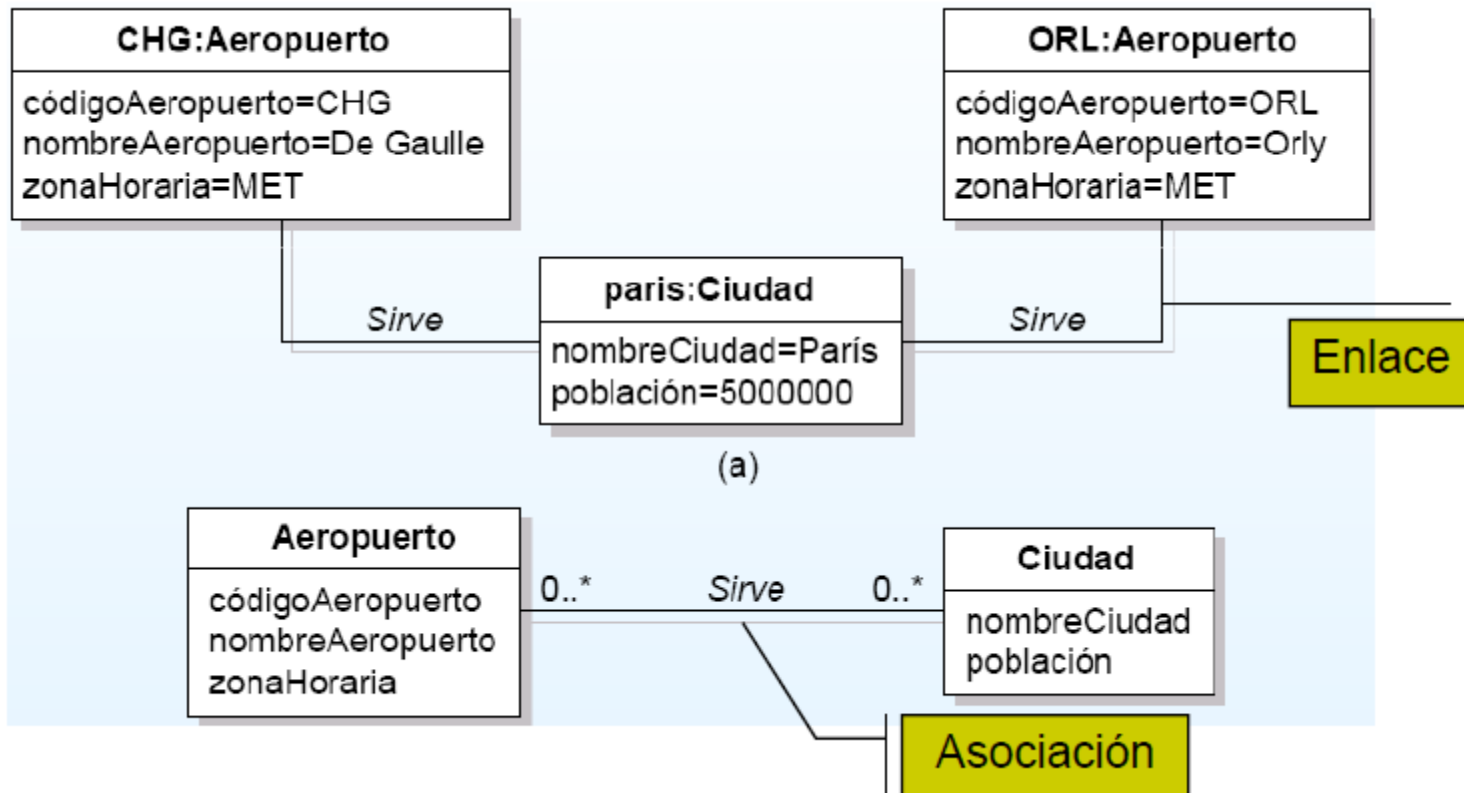
- ▶ Una operación es una función o procedimiento que puede ser ejecutada por un objeto o aplicada a un objeto
- ▶ Cada operación actúa sobre un objeto que es su argumento implícito
- ▶ Al nombre de la operación se le puede agregar detalles tales como la lista de argumentos o el tipo de resultado
- ▶ La implementación de una operación para una clase se denomina método.

ArbolBinario	Ciudad	Aeropuerto	Simulación
	nombreCiudad población	códigoAeropuerto nombreAeropuerto zonaHoraria	descripción fechaEjecución convergencia calcular

Operaciones y métodos

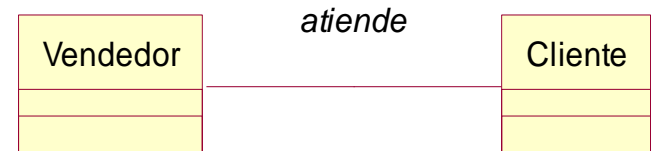
- ▶ Al nombre de la operación se pueden añadir detalles opcionales tales como la lista de argumentos o el tipo del resultado.
 - ▶ Algunas operaciones son **polimórficas**, esto es, aplicables a muchas clases.
 - ▶ La implementación de una operación para una clase concreta se denomina **método**.
- 

Operaciones y métodos



Relación

- ▶ Una relación es una conexión semántica (significativa) entre elementos del modelo.
 - Se le puede dar un nombre, se le puede asignar una cardinalidad y se pueden establecer roles que juegan las clases relacionadas
 - Existen diferentes tipos de relaciones:
 - Herencia,
 - Asociación
 - Agregación
 - Composición

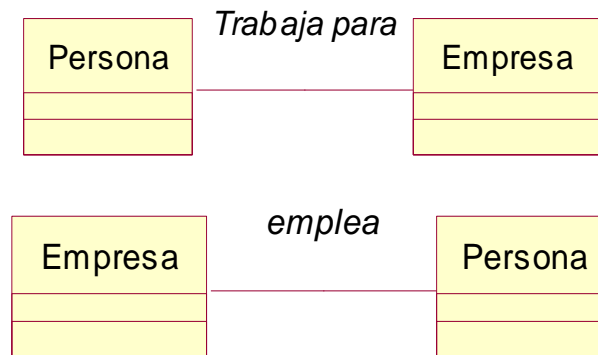


La identificación de relaciones permite reconocer la manera en que colaboran los distintos objetos en el dominio.
Expresan un vínculo **semántico** entre clases

Relaciones

► Asociación

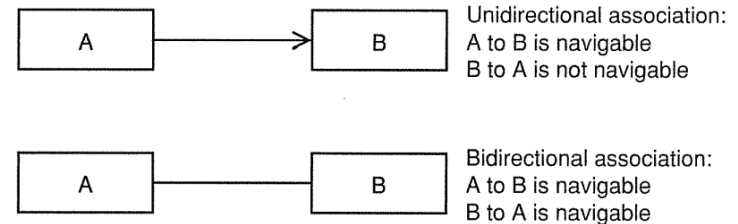
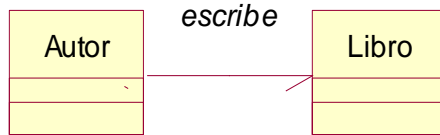
- Relación entre clases.
- Semántica:
 - una asociación entre clases indica que se puede tener vínculos entre objetos de esas clases.
- Sintaxis:
 - Un nombre de asociación.
 - Nombres de rol
 - Multiplicidad
 - Navegabilidad



Relaciones

► Navegabilidad:

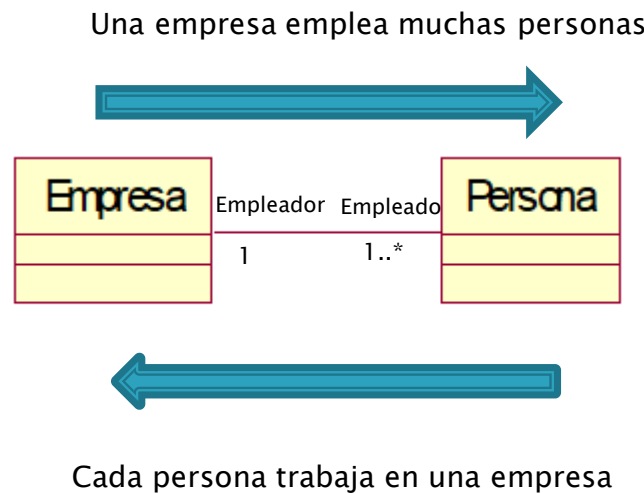
- Es posible pasar de un objeto de la clase origen a uno o varios objetos de la clase destino dependiendo de la multiplicidad.
- Los mensajes sólo pueden ser enviados en el sentido de la flecha.



Relaciones

► Multiplicidad:

- Especifica la cantidad de objetos que pueden participar en una relación en cualquier momento.

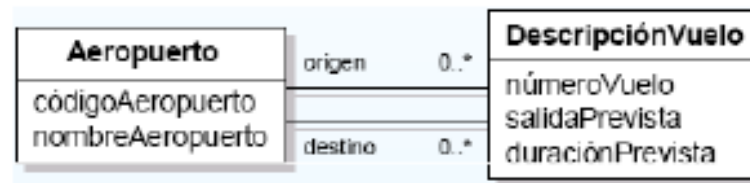


Cardinalidad	Descripción
1	Exactamente 1
*	Cantidad ilimitada
0..*	Cero o más
1..*	Uno o más
0..1	Cero o uno
2..6	Rango específico
2..6, 10	Rango específico o número exacto

Relaciones

► Rol

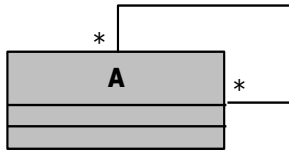
- El rol de un objeto indica el papel que juega en su relación con otros objetos.
- Los nombres de los roles enriquecen la semántica de una relación.
- Son opcionales y aparecen como sustantivos.
- Se escribe cerca de la clase a la cual califica.



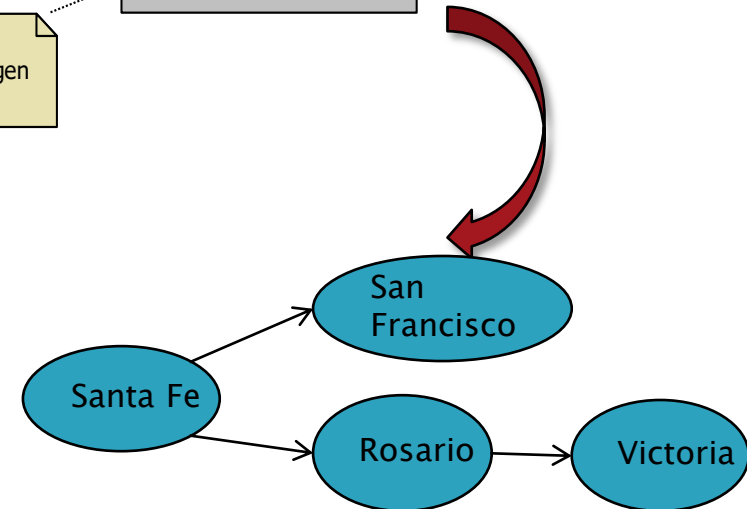
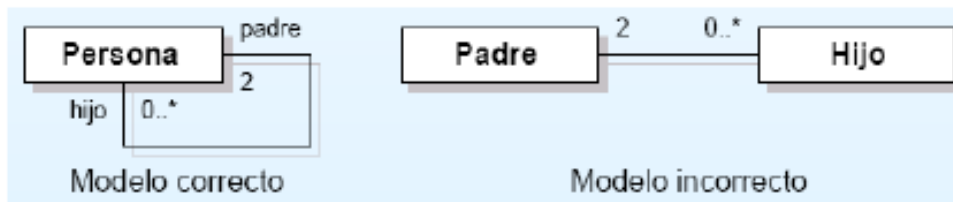
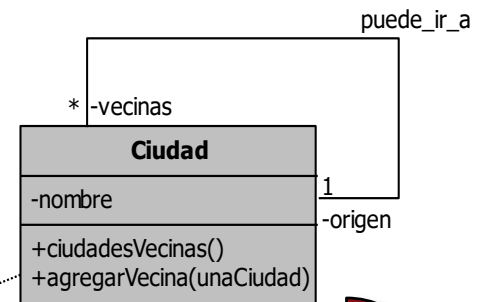
Relaciones

► Asociación reflexiva

- Objetos de la clase tienen vínculos con objetos de la misma clase



Devuelve las ciudades vecinas, desde una origen a los posibles destinos.



Relaciones

► Agregación

- Relación no simétrica
- Permite modelar una relación parte-todo en la cual un objeto es propietario de otro objeto pero no en exclusividad.
- Uno de los extremos cumple un rol predominante respecto del otro extremo.
- No realiza ninguna afirmación sobre el ciclo de vida de las partes involucradas.



Relaciones

► Agregación

◦ Semántica:

- El conjunto puede existir independientemente de las partes.
- Las partes pueden existir independientemente del conjunto.
- El conjunto está incompleto si falta alguna de las partes.
- Es posible tener propiedad compartida de las partes por varios conjuntos.



Relaciones

► Composición

- Relación no simétrica
- Permite modelar una relación parte-todo en la cual un objeto es propietario de otro objeto **en exclusividad**.
- Uno de los extremos cumple un rol predominante respecto del otro extremo.

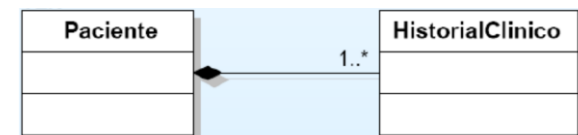


Relaciones

► Composición

◦ Semántica:

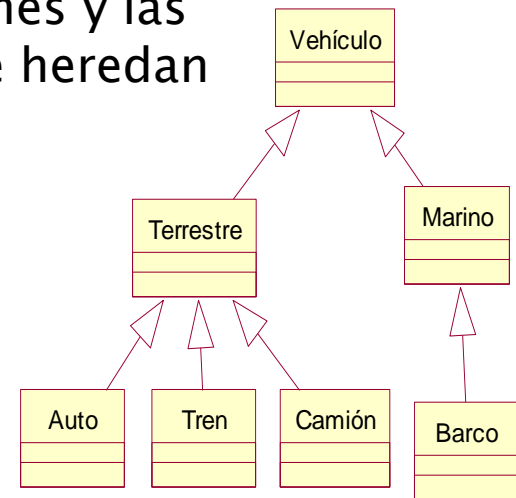
- Las partes con multiplicidad variable pueden ser creadas luego del todo
 - una vez creadas no podrán persistir si el todo desaparece.
- Expresa una relación de pertenencia
- Asocia los ciclos de vida del todo y la parte.
- Las partes pueden solamente pertenecer a un conjunto.
- El conjunto tiene responsabilidad única para las disposición de todas sus partes
 - Tiene la responsabilidad para su creación y destrucción
- Si se destruye el conjunto, debe destruir todas sus partes.



Relaciones

► Generalización

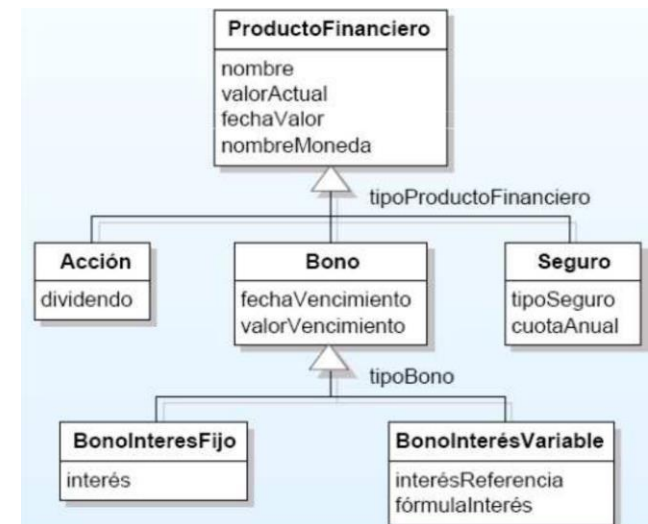
- Relación “es un”.
- Expresa clasificación entre un elemento más general y un elemento más específico.
- Es la relación entre una clase (superclase) y una o más variaciones de esta clase (subclases).
- Una instancia de una subclase es instancia de las superclases de esa clase.
- Los atributos, las operaciones, las relaciones y las restricciones definidas en la superclase se heredan íntegramente en la subclase



Relaciones

► Herencia

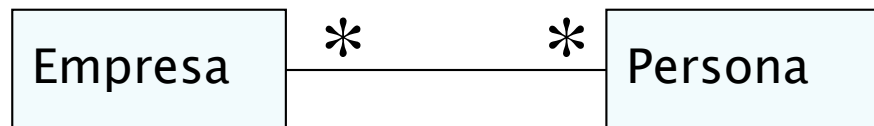
- La generalización es la relación estructural que permite la existencia del mecanismos de herencia.
- Las propiedades heredadas pueden reutilizarse o redefinirse en la subclase.
- La subclase puede definir nuevas relaciones no presentes en la superclase.
- El caso en que una subclase tiene múltiples superclases inmediatas se denomina herencia múltiple.



Relaciones

► Clase asociación

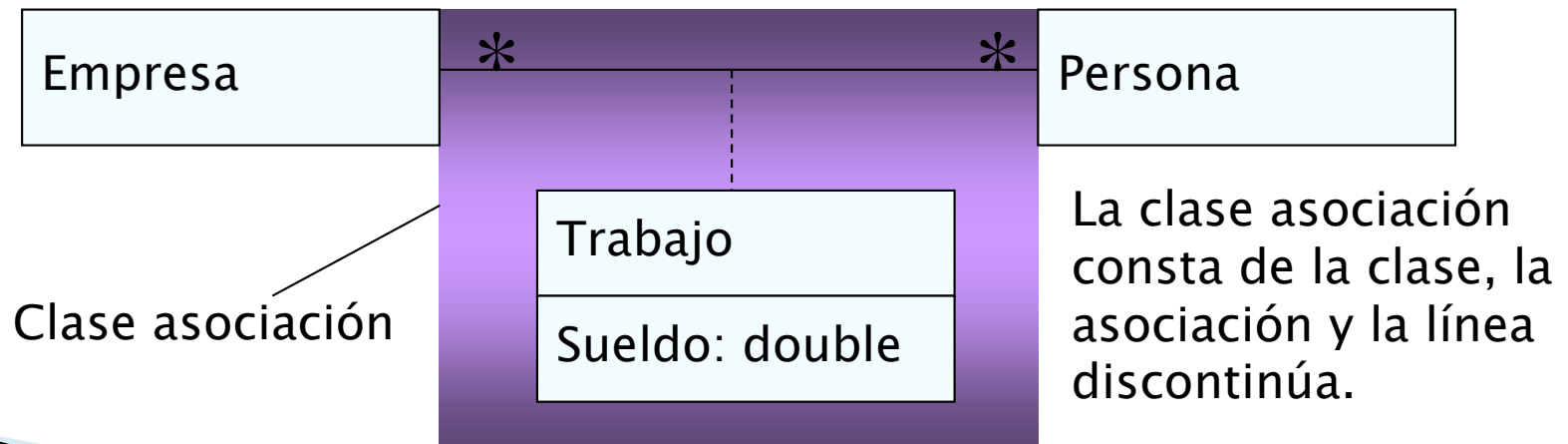
- Es una asociación cuyos enlaces pueden participar en asociaciones posteriores.
 - Tiene características de asociación y de clase:
 - Como el enlace de una asociación, las instancias de una clase de asociación obtienen su identidad de las instancias de las clases que constituyen.
 - Como una clase, puede participar en asociaciones.
- Relación muchos a muchos entre dos clases, donde existen atributos que no se pueden acomodar fácilmente en ninguna de las clases.
- Ejemplo:
 - Todo objeto Persona puede trabajar para muchos objetos Empresa.
 - Todo objeto Empresa puede emplear muchos objetos Persona



Relaciones

► Clase asociación

- Si añade la regla de negocio que cada Persona tiene un sueldo por cada Empresa en la que está empleado.
 - Dónde se debería grabar el sueldo, en la clase Persona o en la clase Empresa?
 - La respuesta es que el sueldo es una propiedad de la propia asociación.
- Para cada asociación de empleo que un objeto Persona tiene como un objeto Empresa, existe un sueldo específico.



Responsabilidad

- ▶ Es una declaración general sobre un objeto de software:
 - una acción que realiza el elemento
 - un conocimiento que mantiene sobre algo
 - una decisión importante que afecta a otro objeto de software.
 - servicio que una clase ofrece
 - contrato u obligación que tiene una clase con sus clientes (objetos que piden un servicio)

Modularidad

▶ Cohesión:

- Es la medida de la fuerza funcional de un módulo o clase.
- Se busca que el módulo tenga la cohesión mas alta posible.
 - Todos sus elementos contribuyen a la ejecución de una misma tarea.

▶ Acoplamiento:

- Es la medida de la interdependencia relativa entre clases o módulos.
- Se busca que exista el mínimo posible de acoplamiento entre módulos
 - Los módulos se comunican solamente por medio de mensajes.

Buena Clase

- ▶ Nombre:
 - refleja su intención
- ▶ Abstracción:
 - modela un elemento específico del dominio
- ▶ Responsabilidades bien definidas
 - Alta cohesión:
 - Concordancia de las responsabilidades con el propósito de la clase
 - Todas las responsabilidades trabajan por el mismo objetivo.
 - Bajo acoplamiento:
 - Reducir el número de clases con las que se relaciona.
 - Distribución uniforme de las responsabilidades entre clases.
 - No localizar el control o muchas responsabilidades en una sola clase.

Pilares de OO

- ▶ Encapsulamiento
- ▶ Herencia
- ▶ Polimorfismo

Polimorfismo

- ▶ Una operación polimórfica tiene muchas implementaciones.

