Análisis Orientado a Objetos en UML

TEMA 2

Departamento de Lenguajes y Sistemas Informáticos

Tema 2. Análisis Orientado a Objetos en UML

Objetivos específicos (2.6)

El alumno debe ser capaz de:

- Describir el objetivo del **Modelo Conceptual de Datos**
- Describir el objetivo del **Diagrama de Clases Conceptuales**
- Definir qué es un objeto y una **clase de objetos**
- Definir qué es un **atributo de una clase de objetos**
- Definir qué es una **asociación entre clases de objetos**
- Definir los siguientes conceptos relacionados con la asociación: **rol, multiplicidad, enlace y atributo de enlace**
- Diferenciar **rol como extremo de una asociación** y **rol como concepto**
- Poner y explicar ejemplos en los que un atributo de enlace se pueda incluir en algunas de las clases de la asociación y ejemplos en los que no se pueda (aunque siempre suponga pérdida de semántica)
- **Describir los conceptos clase de asociación, clasificación y asociación calificada**

Objetivos específicos (2.6)

El alumno debe ser capaz de:

- Describir y diferenciar los conceptos **agregación** y **composición**
- Describir los conceptos **generalización/especialización**
- Enumerar y describir las **restricciones semánticas de la generalización**
- Explicar las **reglas Es-Un** y la **regla del 100%**
- Enumerar las **razones para particionar una clase conceptual en subclases**
- Enumerar las **razones para generalizar subclases**
- Definir los conceptos **herencia múltiple y clasificación múltiple**
- Definir qué es **información derivada (atributo derivado y asociación derivada)**
- Definir los conceptos **multiplicidad de clase y cambiabilidad**
- Definir qué es una **restricción sobre atributo,** una **restricción sobre asociación (XOR y Subset)** y una **restricción textual**
- **Realizar modelos conceptuales de datos de sistemas propuestos**

Modelo conceptual de datos

- Modela los requisitos de datos de un sistema.
- Se describen las estructuras de datos de un sistema y las relaciones estáticas que existen entre ellos.

> *Es la representación de los conceptos (objetos) significativos en el dominio del problema.*

- Normalmente contienen:
  - Clases de objetos.
  - Asociaciones entre clases de objetos.
  - Atributos de las clases de objetos.
  - Restricciones de integridad.
- Técnica: Diagrama de Clases

Objetos

- **Objeto:**
  - Entidad que existe en el mundo real
  - Tienen identidad propia y son distinguibles entre ellos
- **Ejemplos:**
  - El avión con matrícula 327
  - El avión con matrícula 999
  - La factura 3443
  - Una manzana
  - Una mesa
  - Un ordenador con nº inventario C-1122
  - Etc.

Clases de Objetos

- **Clase de Objeto:** Describe un conjunto de objetos con:
  - las mismas propiedades
  - Comportamiento común
  - Idéntica relación con otros objetos
  - Semántica común

- Avión con matrícula 327
- Avión con matrícula 999

abstracción →

Avión

Eliminar distinciones entre objetos para poder observar aspectos comunes

Clase Avión

*Los objetos de una clase tienen las mismas **propiedades** y los mismos **patrones de comportamiento***

Clases de Objetos

- Ejemplo: Venta de Productos

"Un terminal de punto de venta (TPV) es un sistema que se usa para registrar las ventas de productos a clientes y para gestionar los pagos. Se usa principalmente en supermercados y grandes superficies. Incluye componentes hardware (ordenador y escáner del código de barras) y software para ejecutar el sistema".

• Clases de Objetos

TPV | Supermercado | Venta | Línea de Venta

Cliente | Producto | Pago

Atributos

- **Atributo:**
  - Propiedad compartida por los objetos de una clase

TPV
num-pv: Integer

Supermercado
dirección: String nombre: String

Venta
fecha: Date hora: Time

Línea de venta
cantidad: Integer

Cliente
nombre: String telfs [1..*]:Integer tipcli: TipoCliente

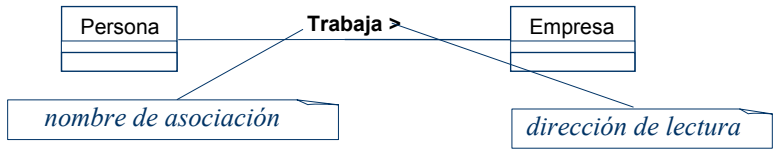
Producto
upc: Integer descripción [0..1]: String Precio: Integer

Pago
importe: Integer

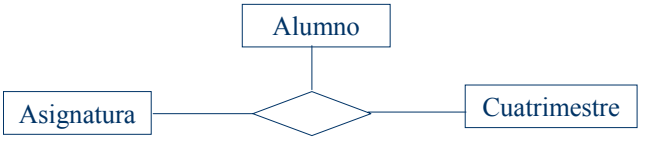
- Pueden tomar valores nulos (ej. *descripción*)
- Pueden ser multivaluados (ej. *telfs*)
- Pueden ser definidos por el usuario mediante enumeraciones
- TipoCliente se define como una enumeración con los valores que puede tener

Asociaciones

- **Asociación**
  - Es la representación de relaciones entre dos o más objetos



  - La navegación de una asociación por defecto es bidireccional
  - Al nombrar una asociación hay que elegir una determinada dirección
- **Asociación de orden superior a dos**

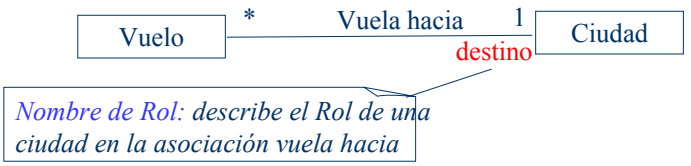


Cada instancia de la asociación es una n-tupla de valores de cada una de las respectivas clases (cardinalidad)

Asociaciones

Si una clase C puede tener simultáneamente muchos valores para el mismo tipo de atributo A, no coloque el atributo A en C. Coloque el atributo A en otra clase que esté asociada con C.

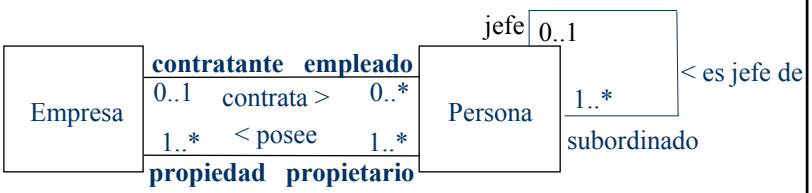
- **Nombre de Rol en las asociaciones**
  - Cada extremo en una asociación es un rol, que tiene diversas propiedades como el *nombre* y la *multiplicidad*.
  - El nombre de rol identifica un extremo de la asociación y describe el papel que desempeñan los objetos en la asociación.



Rol

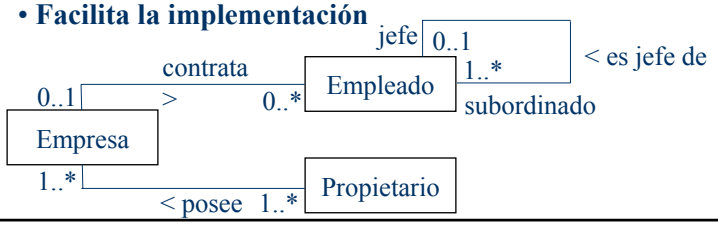
- En ocasiones un rol puede modelarse de dos formas (especialmente un rol humano)

• **Rol en asociación**



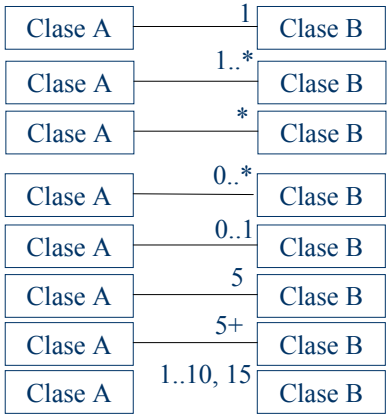
• **Rol como conceptos**

- **Facilita inclusión:** Atributos únicos, Asociaciones, Semántica
- **Facilita la implementación**



Multiplicidad de las asociaciones binarias

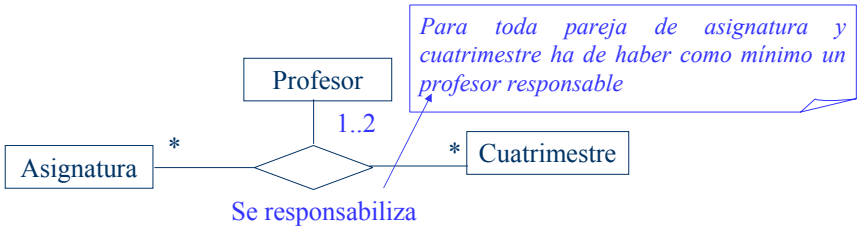
Dada una instancia *a* de la clase A cualquiera, la multiplicidad del extremo B define cuántas instancias de B se pueden asociar con *a* en un momento de tiempo determinado

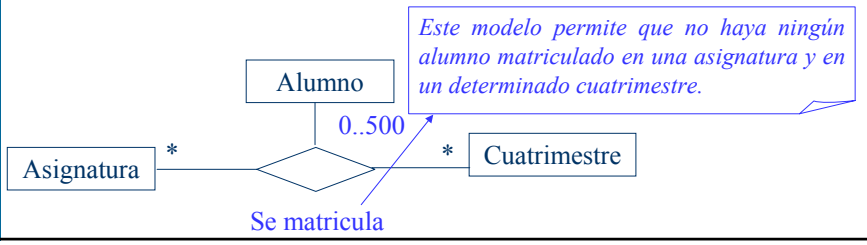


Cota inferior: n+
Cota cjto.: n_1, n_2...
Cota rango: m..n

Multiplicidad en las asociaciones ternarias

Dadas una instancia *a* de A y una instancia *b* de B cualesquiera, la multiplicidad en el extremo C nos dice cuántas instancias de C se pueden asociar con la pareja (*a*,*b*).

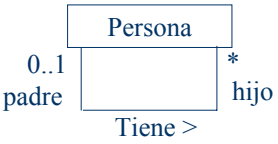


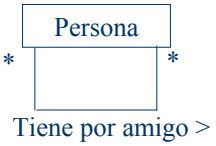


Asociaciones

- **Asociaciones recursivas**

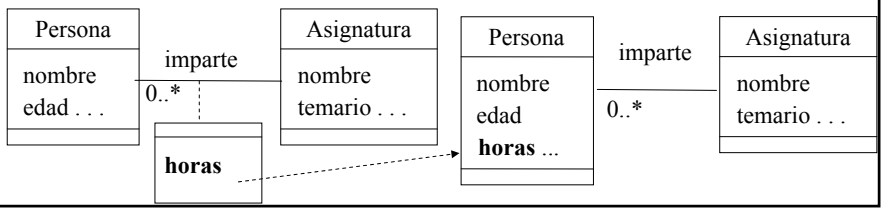
  Asociaciones en las que una misma clase de objetos participa más de una vez (con papeles diferentes o no)



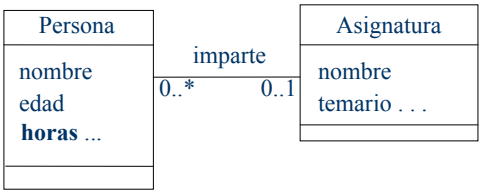


Asociaciones

- **Enlace:** Instancia de una asociación
- **Atributos de enlace.**
  - Propiedad de los enlaces de una asociación.
  - No se pueden asociar a ninguno de los objetos que intervienen en la asociación sin perder información.
  - ¿Se pueden **trasladar atributos de enlace** a alguna de las clases de la asociación?:
    - **SÍ** en asociaciones *1:1* ó *1:m.* Se trasladan a la clase del extremo contrario al de *multiplicidad 1.*
    - **NO** en asociaciones *n:m.*

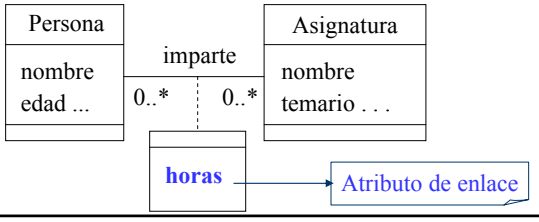


Atributos de enlace



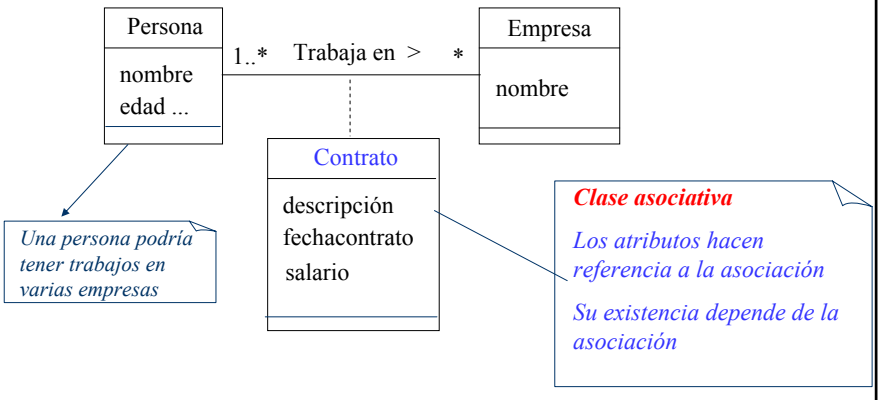
• **Persona:** incluye a <u>NO docentes</u>

- **¿HORA es un atributo de persona?**
  - ¿Qué ocurre con las personas que no dan clase con el *nº horas*?.
  - ¿Qué ocurre si queremos contemplar que una persona imparta *n asignaturas*?



Clase Asociativa

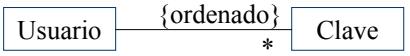
- **Clase asociativa (clase de asociación)**
  - Representar una **asociación** como una **clase** (con atributos y operaciones).
  - Una clase asociativa puede participar en otras relaciones



Clase de Asociación y Clasificación

- **Guías para incluir clases de asociaciones**
  - Un atributo está relacionado con una asociación
  - El tiempo de vida de las instancias de la asociación depende de la asociación
  - Existe una asociación *n:m* entre dos clases e información asociada con la propia asociación

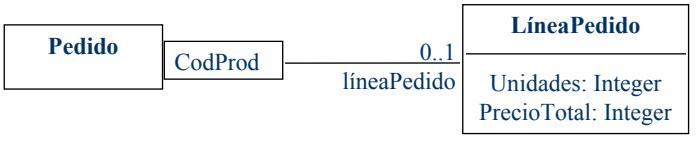
- **Clasificación (restricción semántica asociación)**
  - Objetos del lado *muchos* de la asociación tienen un orden explícito



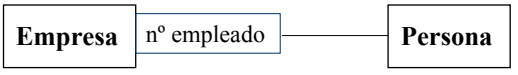
Las claves asociadas con el usuario podrían mantenerse en orden de menos a más recientemente usada

Asociación Calificada

- **Asociación calificada o Calificación (precisión semántica)**
  - Relaciona dos clases (asociación *1:m* ó *n:m*) y un calificador.
  - **Calificador:** Distingue entre el conjunto de objetos del lado muchos (reduce multiplicidad de la asociación)



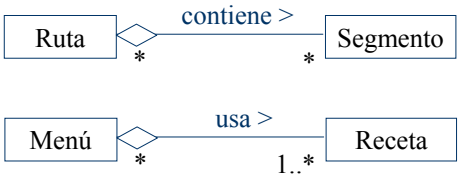
"*Dentro del mismo pedido no pueden existir dos líneas con el mismo producto*". Pueden distinguirse las líneas de pedido en un pedido según su CodProd.



"*En una determinada empresa no pueden existir dos empleados con el mismo número*". Pueden distinguirse los empleados de una empresa por su número de empleado.

Agregación

- **Agregación** (propiedad de un Rol)
  - Es un tipo de asociación usada para modelar relaciones "*parte-todo*" entre objetos.
  - El "*todo*" se denomina **compuesto** y las "*partes*" **componentes**.



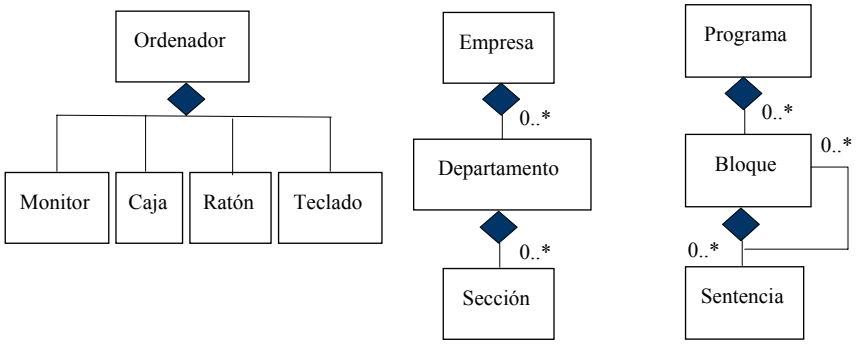
- La distinción entre asociación y agregación es a menudo subjetiva.

Agregación

Fija
Niveles fijos y Partes fijos

Variable
Niveles fijos y Partes variable

Recursiva
Niveles ilimitado

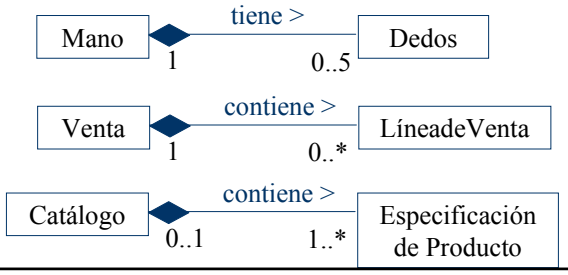


Según la dependencia y la exclusividad de la agregación se diferencian cuatro tipos de agregaciones:

- **Dependencia:** *¿La existencia de una parte va ligada a la del agregado?*
- **Exclusividad:** *¿Una parte puede pertenecer a más de un agregado?*

Composición

- **Composición**
  - Es un tipo de agregación **exclusiva y dependiente**
  - La multiplicidad del extremo compuesto puede ser como máximo 1 (como máximo un componente lo es de un compuesto)
  - Si un componente está asociado a un compuesto y el compuesto se borra entonces el componente también se ha de borrar (no lo puede sobrevivir).
  - Las partes se pueden borrar antes de borrar el compuesto.



Agregación y Composición

¿Cuándo mostrarla?

Agregación/Composición

- Existe una relación todo-parte física o lógica
- Algunas propiedades del compuesto se propagan a los componentes (destrucción, movimiento…)

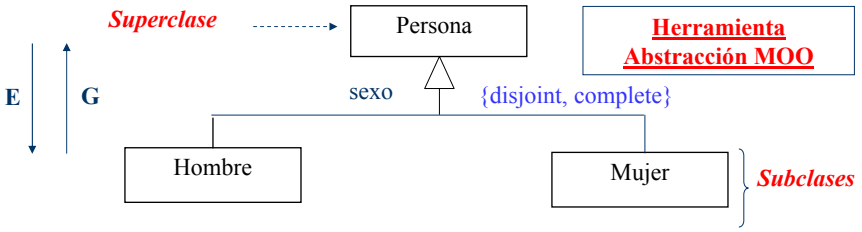
Composición

- La vida del componente depende de la vida del compuesto
- Existe una dependencia crear-borrar del componente respecto del compuesto

- Si hay duda, descartarla
- Beneficios representar agregaciones: Diseño

Generalización/Especialización

Identificar elementos comunes entre los objetos definiendo relaciones de superclase (objeto general) y subclase (objeto especializado)



Discriminador: Es el nombre de la partición. Ha de ser único entre los atributos y roles de la superclase

Restricciones semánticas:

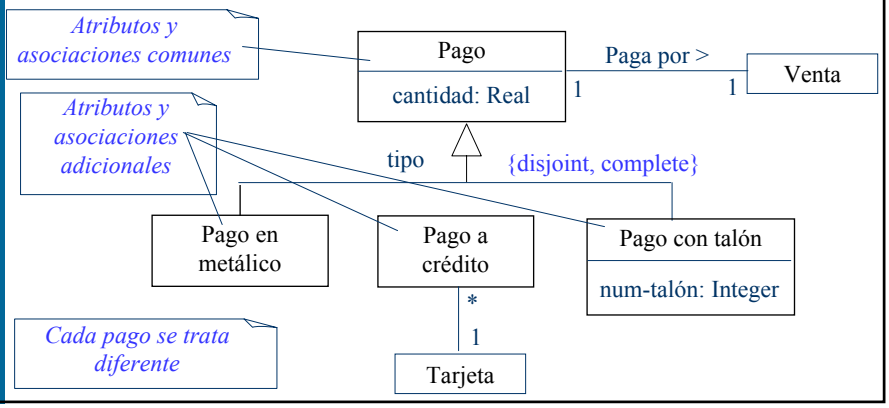
- **disjoint:** Un descendiente no puede ser de más de una subclase.
- **overlapping:** Un descendiente puede ser de más de una subclase.
- **complete:** Se han especificado todas las subclases.
- **incomplete:** La lista de subclases es incompleta.

Generalización/Especialización

- **Regla del 100%**
  - El 100% de la definición de la superclase se debe poder aplicar a la subclase. La subclase debe ajustarse al 100% de los: *Atributos*, *Asociaciones y Restricciones de Superclase*
- **Regla Es-Un**
  - Todos los objetos de una *subclase* deben ser objetos de su *superclase*
- **Subclase potencial debe estar de acuerdo:**
  - Regla del 100 % (conformidad con la definición)
  - Regla Es-Un (conformidad con pertenencia a superclase)

Generalización/Especialización

- **Razones particionar clase conceptual en subclases**
  - La subclase tiene atributos adicionales
  - La subclase tiene asociaciones adicionales
  - La subclase es tratada o manipulada de manera diferente a la superclase o a otras subclases



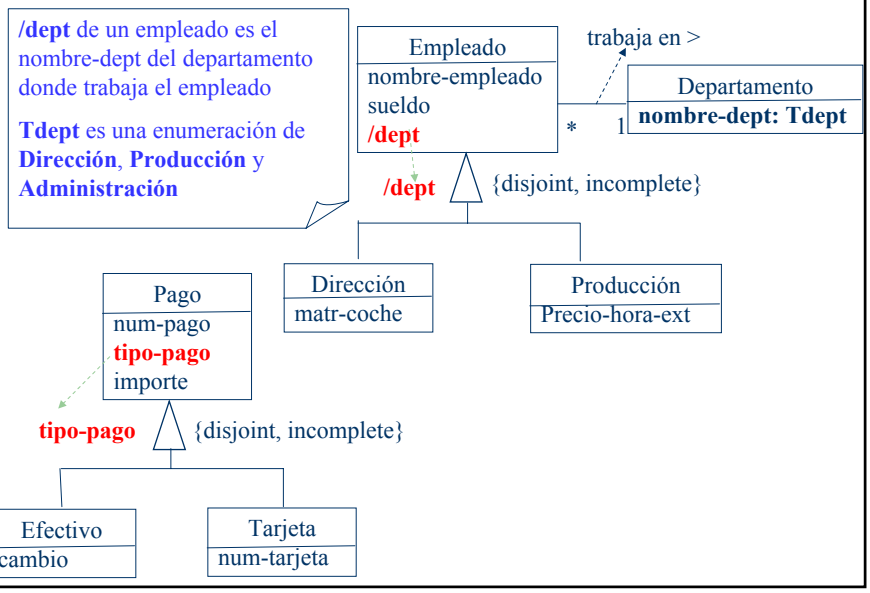
Generalización/Especialización

- **Razones para definir una superclase (generalizar subclases)**
  - Las subclases potenciales representan variaciones de un mismo concepto
  - Las subclases tienen atributos que pueden ser factorizados y expresados en las superclases
  - Las subclases tienen asociaciones que pueden ser factorizadas y relacionadas con la superclase
  - Las subclases se ajustan a las *reglas 100%* y *Es-Un*

- **Ver ejemplo transparencia anterior**

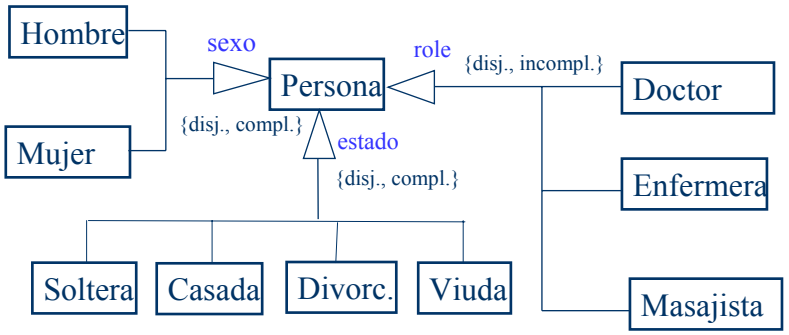
Generalización/Especialización

Ejemplos discriminador



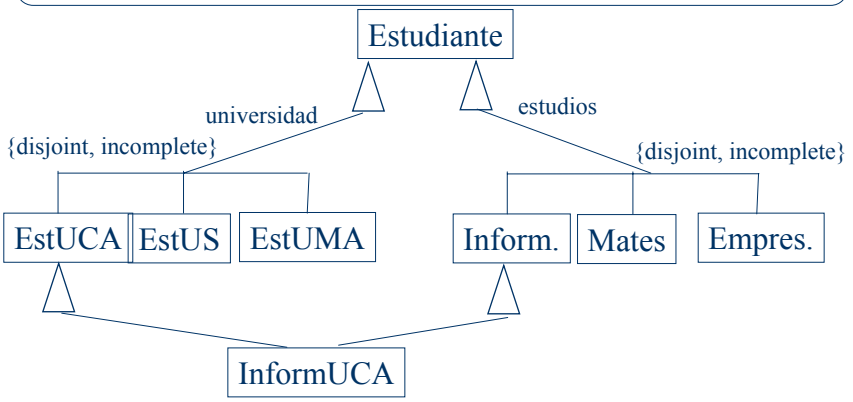
Clasificación Múltiple

Variante de generalización/especialización en la cual una superclase puede tener diversas jerarquías de especialización en función de diferentes discriminadores



Herencia Múltiple

Variante de generalización/especialización en la cual una subclase tiene más de una superclase



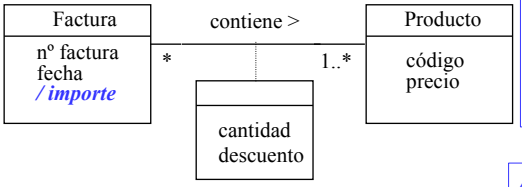
Sólo se puede utilizar si no hay **conflictos de herencia**

Conflicto de herencia: Una clase tiene más de una superclase con un mismo atributo/asociación (operación) que no proviene de un único antecesor

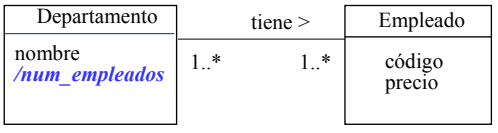
Información derivada

- Un elemento (atributo o asociación) **es derivado** si se puede obtener a partir de otros elementos (no añaden información fundamental).
- Se incluye cuando mejora la **claridad** del modelo conceptual
- Una ***constraint*** (regla de derivación) ha de especificar cómo se deriva

- **Atributo derivado**



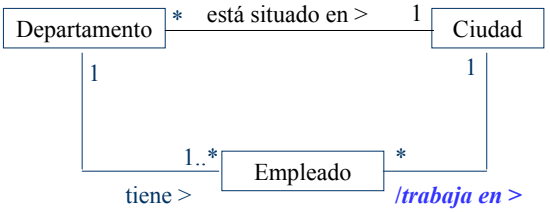
/importe: *El importe de una factura* ***f*** *es igual a la suma del importe de los productos (cantidad x precio) que contiene la factura* ***f****.*



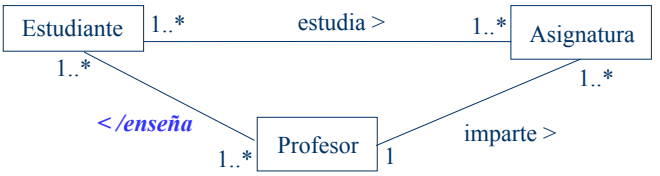
/num_empleados: *El número de empleados de un departamento* ***d*** *es igual al número de ocurrencias de* ***tiene*** *donde aparece* ***d***

Información derivada

- **Asociación derivada**



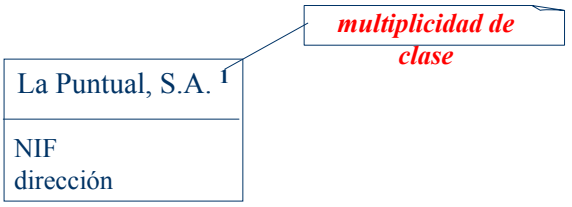
/trabaja en: *La ciudad donde trabaja un empleado es la ciudad donde está situado su departamento.*



/enseña: *Los estudiantes a los que enseña un profesor son los estudiantes que estudian las asignaturas que imparte el profesor*

Multiplicidad de clase

- La **multiplicidad de clase** establece el rango de posibles cardinalidades para las instancias de una clase.
- Por defecto, es **indefenida.**
- En algunos casos es útil establecer una multiplicidad finita, especialmente en casos de clases que pueden tener una sóla instancia (y que se denominan *singleton*).



Cambiabilidad

- La **cambiabilidad** indica si los valores de un atributo o el extremo de una asociación pueden cambiar o no.

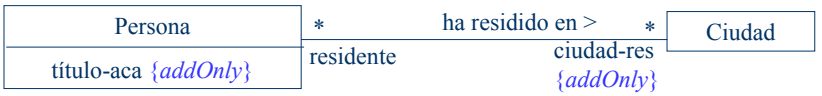
• **Cambiable (*changeable*)**

teléfono y ciudad-res son "cambiables"

• **Congelado (*frozen*)**

fecha-nac y ciudad-nac son "congelados"

• **Sólo añadir (*addOnly*)**

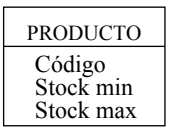


título-aca y ciudad-res son "sólo añadir"

Restricciones

- **Restricciones**
  - Limitan los valores que pueden tomar las clases, los atributos o las asociaciones

- **Restricciones sobre atributos**



{Stock max > Stock min}
{Stock max = 1000}
{Stock min > = 10}

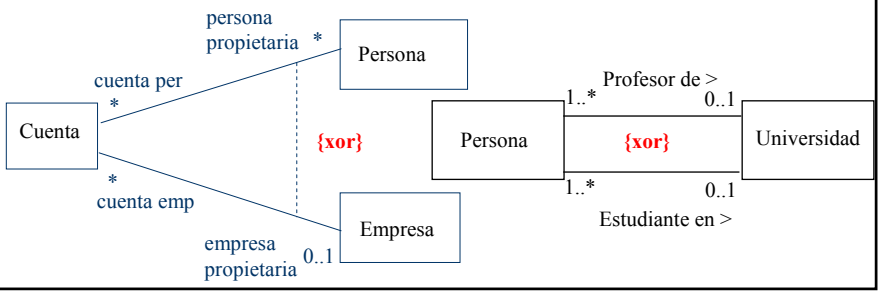
Restricciones

- **Restricciones sobre asociaciones**

  A parte de la multiplicidad, es posible expresar otras restricciones sobre las asociaciones:

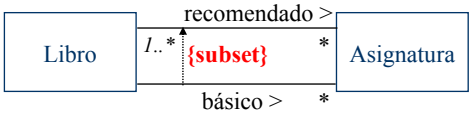
  - **Xor**
  - **Subset**

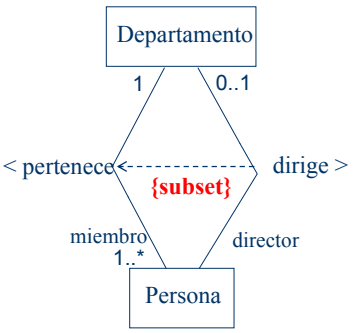
- **Xor**
  - Une diversas asociaciones ligadas a una misma clase base
  - Una instancia de la clase base puede participar como máximo en una de las asociaciones unidas por ***xor***



Restricciones

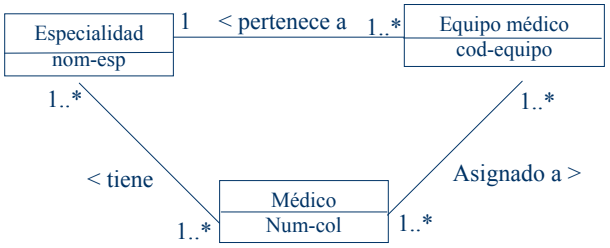
- **Subset**
  - Indica que una asociación es un subconjunto de otra asociación





Restricciones textuales

- Las restricciones que no se pueden especificar gráficamente con la notación UML se especifican de forma textual.
- La especificación textual se puede hacer con lenguaje natural, con OCL, etc.



1. *Dos especialidades diferentes no pueden tener el mismo nom-esp.*
2. *Dos equipos médicos diferentes no pueden tener el mismo cod-equipo.*
3. *Dos médicos diferentes no pueden tener el mismo num-col.*

Restricciones de clave externa

4. *Un médico no puede estar asignado a un equipo médico que pertenezca a una especialidad que el médico no tiene*

Ejemplos Modelo Conceptual de Datos

Realizar un modelo conceptual de datos que responda a las siguientes especificaciones:

Las áreas metropolitanas tienen una serie de hoteles, algunos de los cuales pertenecen a una determinada cadena de hoteles. Los hoteles aceptan varias tarjetas de crédito. De las áreas metropolitanas interesa conocer el nombre del área, el nombre del estado o provincia a la que pertenece y el nombre del país. De los hoteles interesa conocer el nombre, la dirección, el nº de habitaciones, el nº de teléfono, el nº de estrellas, el precio de la habitación simple y el precio de la habitación doble. De las cadenas de hoteles interesa conocer el nombre y el director. De las tarjetas de crédito sólo interesa conocer el nombre.

Ejemplos Modelo Conceptual de Datos

Realizar un modelo conceptual de datos que responda a las siguientes especificaciones de un sistema de vuelos:

> Los aeropuertos dan servicio a varias ciudades y hay ciudades que tienen más de un aeropuerto. Los vuelos entre los aeropuertos los gestionan las líneas aéreas y se describen como se muestra en el siguiente ejemplo: *vuelo TW250* del aeropuerto de *S. Pablo de Sevilla* al aeropuerto *Reina Sofía de Tenerife*, tiene prevista la salida a las *7:42 am* y una duración estimada de *1 hora y 80 minutos*; el vuelo se realizará en un *DC9* (modelo de avión) todos los días de la semana excepto los sábados y estará vigente desde *febrero de 2005* hasta *junio de 2006*. De los modelos de aviones interesa conocer el código del modelo (ej. DC9) y el fabricante, de los aeropuertos el código y el nombre, y de las ciudades y las líneas aéreas sólo interesa conocer el nombre.

Referencias de Modelado Conceptual

- G. Booch, J. Rumbaugh, I. Jacobson, "***El Lenguaje Unificado de Modelado. Guía de Usuario*",** Addison Wesley, 1999.

- G. Booch, J. Rumbaugh, I. Jacobson, "***El Lenguaje Unificado de Modelado. Manual de Referencia*",** Addison Wesley, 1999.

- C. Larman, *"**UML y Patrones: Una introducción al análisis y diseño orientado a objetos y al proceso unificado",*** Cap. 10, 11, 12, 26 y 27 . Prentice-Hall, 2003.