Diagramas de clases

Diagramas de clases

- Los diagramas de clases son las representaciones más utilizadas en el modelado de sistemas orientados a objetos.
- Son representaciones gráficas de la parte estática de un sistema.
- Describe la estructura del sistema mostrando un conjunto de clases, interfaces y colaboraciones, así como sus relaciones.

Diagramas de clases

- Permite representar modelos en distintas etapas de desarrollo con diferentes perspectivas:
  - Modelo de análisis: **Diagramas de clases conceptuales.**
  - Modelo de diseño: **Diagramas de clases de diseño.**
  - Modelo de implementación: **Diagramas de clases de implementación.**

Diagramas de clases: Elementos

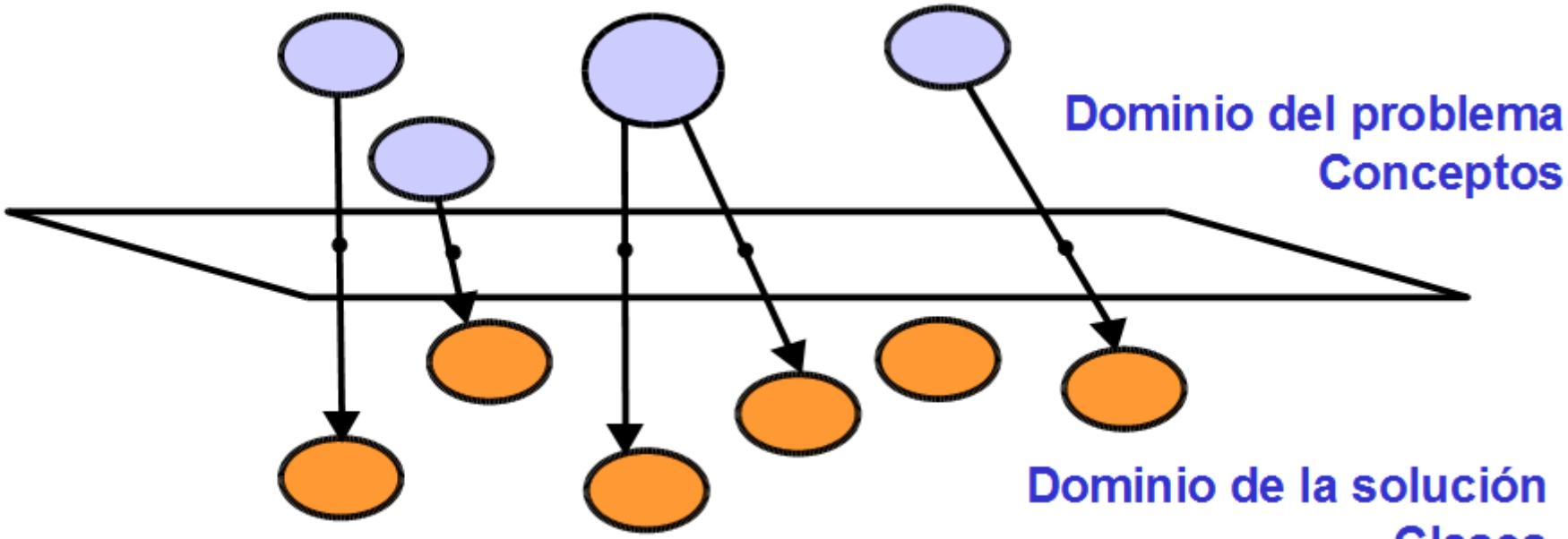
Un diagrama de clases está compuesto por:

- Elementos estructurales: las **clases** y las **interfaces**.
- Relaciones entre esos elementos:
  - **Asociación.**
  - **Generalización.**
  - **Dependencia.**
  - **Realización**.
- Notas y estereotipos.

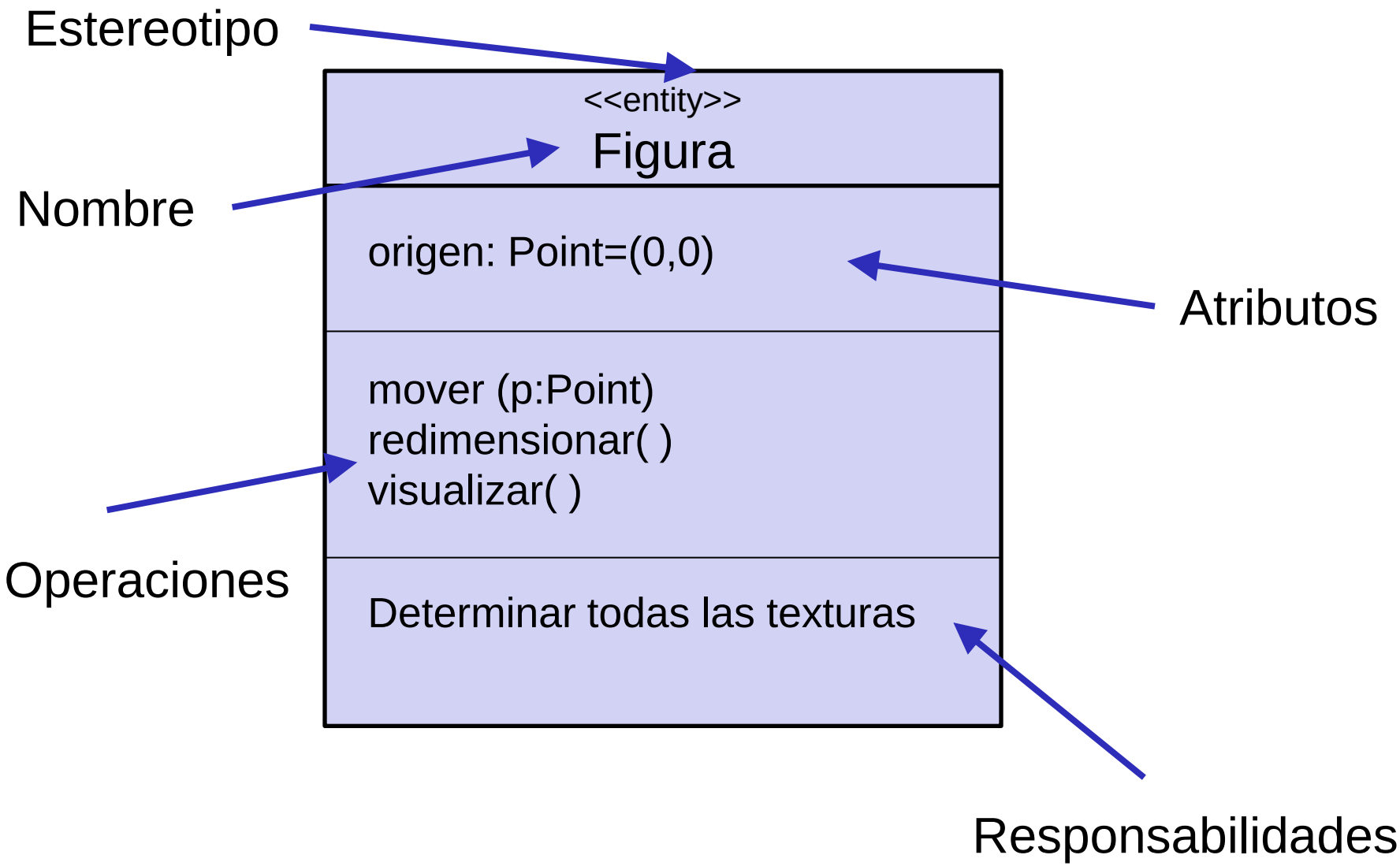
Clases

*"**Una clase** es una descripción de un conjunto de objetos que comparten: atributos, operaciones, relaciones y semántica".*

Define un conceptos que forma parte del dominio del problema o de la solución.



Clases: Notación



Clases: Notación

Nombre: Expresión nominal extraída del vocabulario del problema a modelar. Debe comenzar con mayúscula.

Estereotipo: Permite establecer la categoría de la clase.

Atributo: Propiedad del elemento que se está modelando.

[visibilidad] nombre [multiplicidad] [:tipo] [=valor inicial]

Operación: Es la abstracción de un servicio que puede prestar ese objeto.

[visibilidad] nombre [(lista de parámetros)] [:tipo retorno]

Responsabilidad: Obligación de esa clase con las demás.

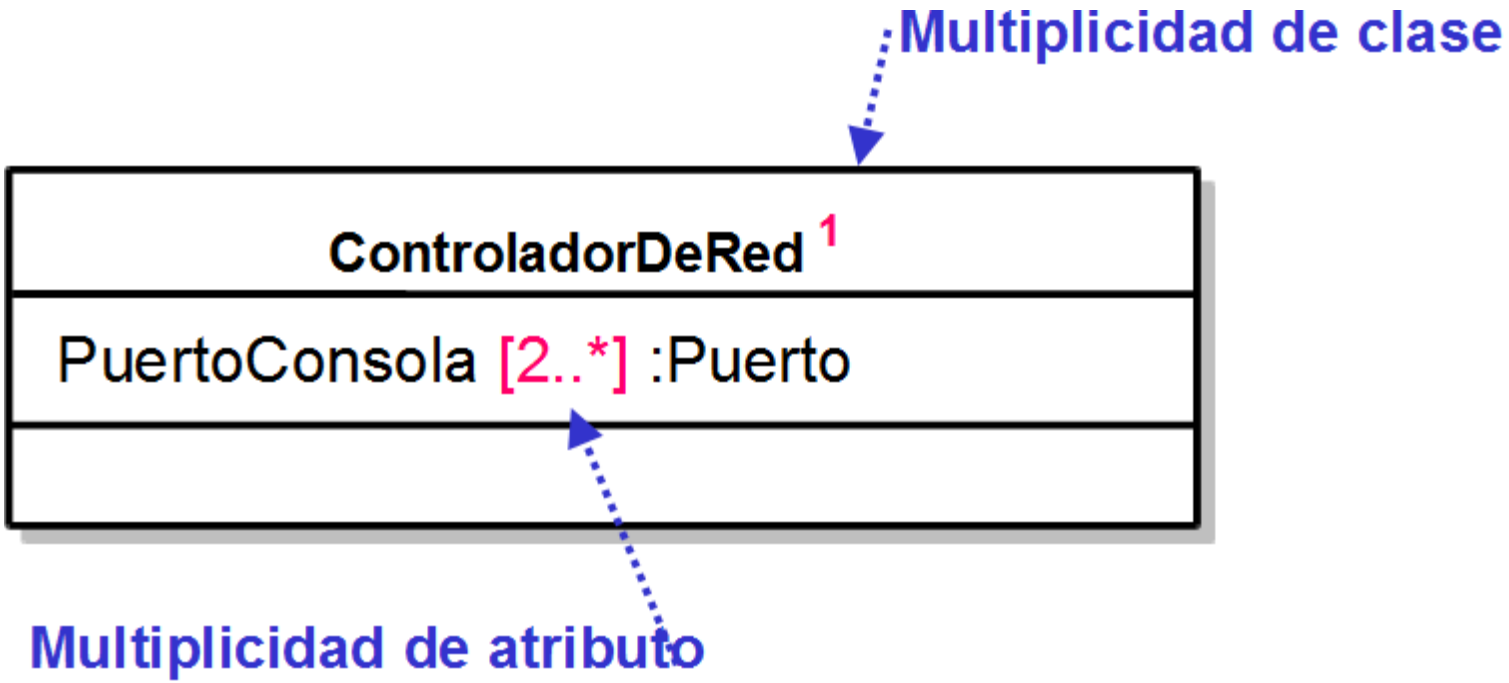
Clases. Encapsulación

- **Visibilidad** de los atributos y operaciones:
  - **Pública**: Cualquier clase puede usar ese atributo u operación. (**+**)
  - **Paquete**: Sólo el paquete en el que está definida la clase puede usar ese atributo u operación. (**~**) (valor por defecto)
  - **Protegida**: Cualquier clase hija puede usar ese atributo u operación. (**#**)
  - **Privada**: Sólo la propia clase puede usar ese atributo u operación. (**-**)
- **Alcance** de los atributos y operaciones:
  - **De instancia**: Cada instancia posee su propio atributo u operación. (valor por defecto)
  - **De clase**: Existe un único valor para todas las instancias de la clase. Se representa subrayando el nombre.

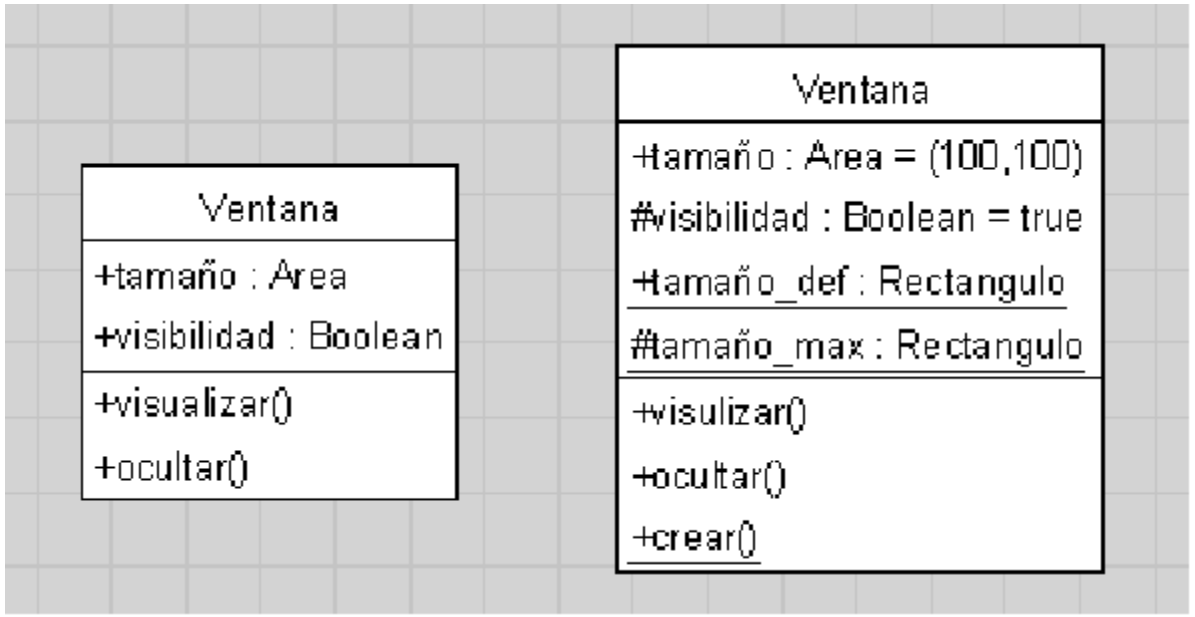
Clases. Multiplicidad

De **clase**: Número de instancias que puede tener una clase.

De **atributo**: Número de instancias de un atributo.

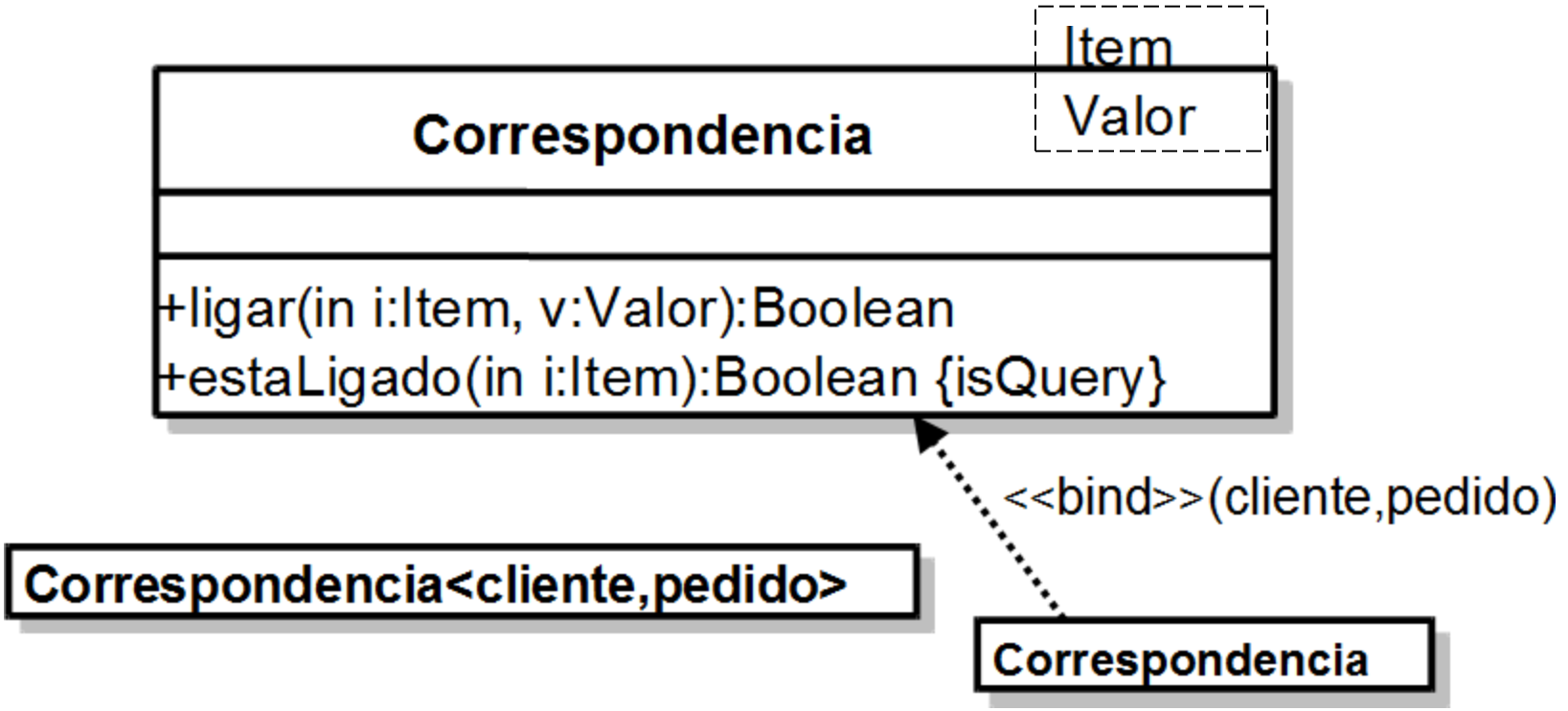


Ejemplos de Clases



Parametrización

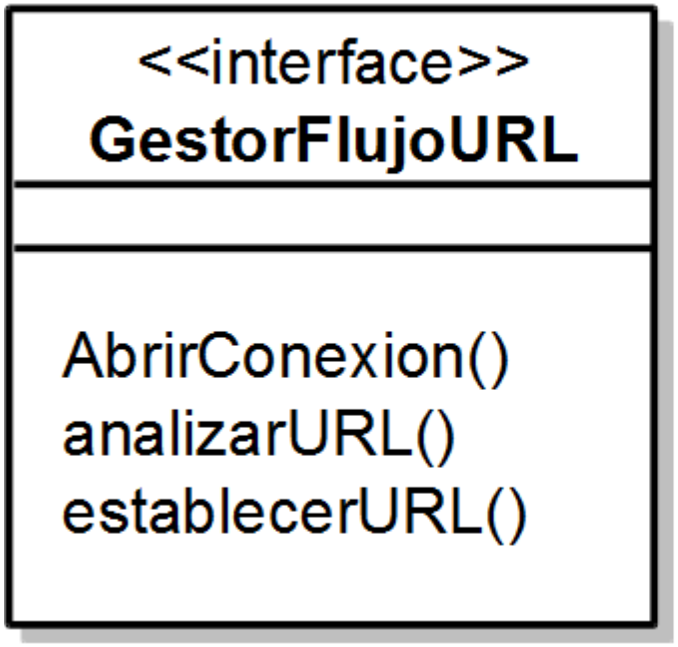
Clase plantilla: Clase con algunas de sus propiedades parametrizadas. Para su uso es necesario que tomen valores los parámetros.



Interfaces

Una **Interfaz** sirve para **especificar el servicio** de las clases. Está compuesta por un nombre y definición de un conjunto de operaciones.

Hay dos representaciones distintas para una interfaz:



Relaciones entre clases

Las **relaciones** modelan la forma en la que los elementos estructurales se conectan entre sí.

Tipos de relaciones:

- *Asociación.*
- *Generalización.*
- *Dependencia.*
- *Realización (o Instanciación).*

Relación de Asociación

Una **Asociación** es una semántica entre dos o más clases que implica conexiones entre sus instancias.

Un **enlace** es una conexión entre objetos y representa una instancia de una asociación entre clases.

Relación de Asociación: Adornos

Los **adornos** son las distintas propiedades de la asociación que modelan su semántica.

Pueden ser:

- Nombre de la asociación.
- Navegabilidad.
- Nombre de rol.
- Multiplicidad.
- Agregación/Composición.
- Cualificador.
- Visibilidad.

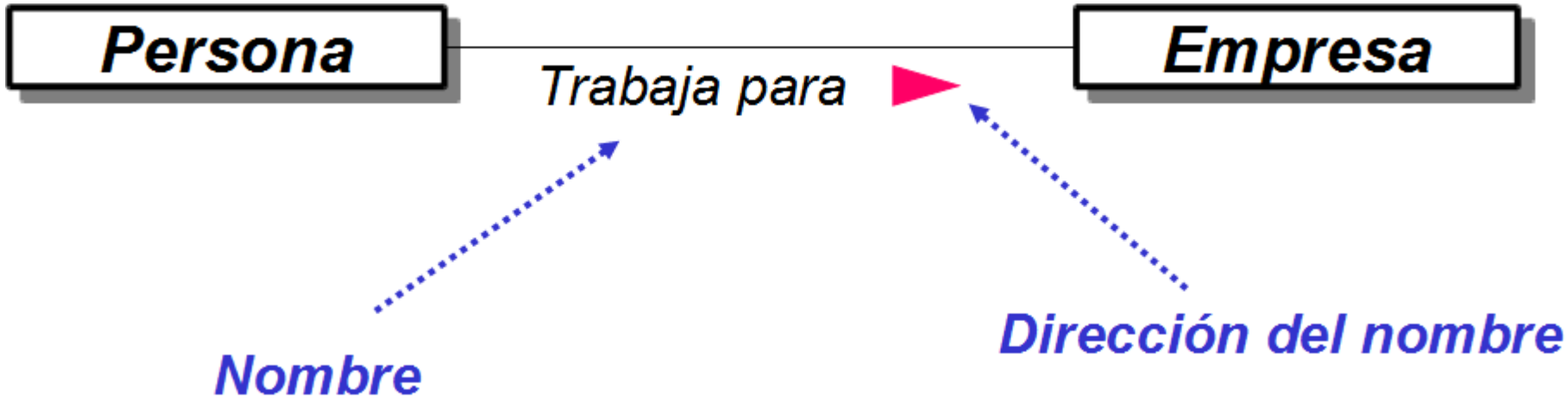
Adornos: Nombre de la asociación

- El **nombre de asociación** identifica la asociación que se establece entre dos clases.

- Los nombres de la asociación deberían ser frases verbales porque indican una acción que objeto fuente está realizando sobre el objeto destino.

Para su representación:

- El nombre de asociación es opcional.
- El nombre de asociación se escribe en minúsculas.
- Puede llevar una flecha que indica el sentido en el que se debería leer la asociación (opcional).

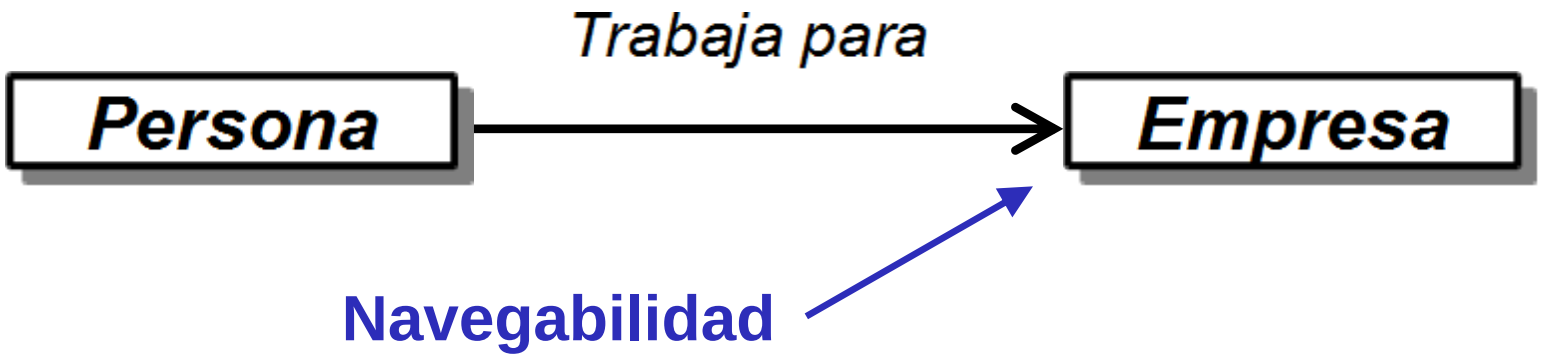


Adornos: Navegabilidad

- En una asociación, se puede señalar la **navegación** desde un objeto de una clase hasta un objeto de la otra clase.

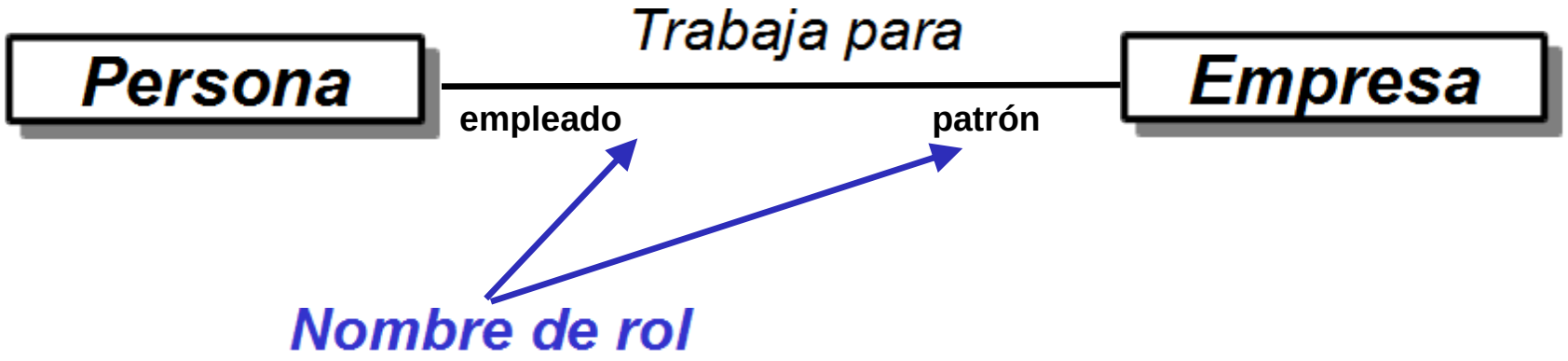
Para su representación:

- La navegabilidad se muestra con una punta de flecha en un extremo de la asociación.
- Cuando la asociación no lleva punta de flecha indicará que la navegabilidad es en los dos sentidos.



Adornos: Nombre de Rol

- El **rol** indican el papel o la cara que la clase de un extremo de la asociación presenta a la clase del otro extremo.
- Los nombres de roles deberían ser nombres o frases nominales ya que nombran un rol que pueden desempeñar objetos.
- Son propios de la asociación.



Adornos: Multiplicidad

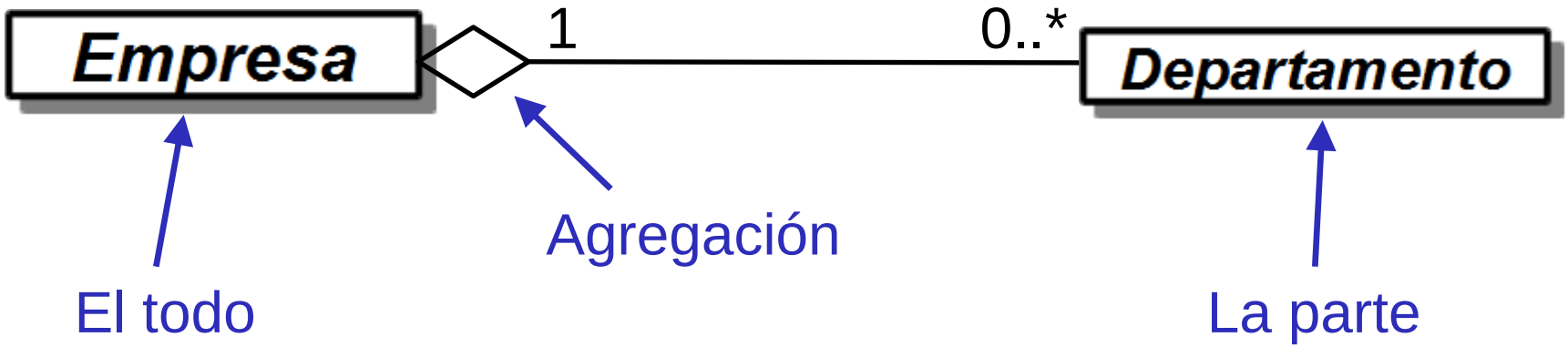
- La **multiplicidad** indica *cuántos* objetos de un extremo de la asociación pueden conectarse con un objeto del otro extremo.

- La multiplicidad restringe el número de objetos de una clase que se pueden implicar un una relación determinada en cualquier momento en el tiempo.

Expresión de la multiplicidad (mínima..máxima)			
1	Uno y solo uno	*	Cero o muchos
0..1	Cero o uno	0..*	Cero o muchos
m..n	Desde m hasta n	1..*	Uno o muchos
0..1,3..4,6..*	Cualquier número excepto 2 y 5		

Adornos: Agregación

*Una **Agregación** (relación "parte-de") es una asociación en la que una de las clases representa el "todo" y la/s otra/s la/s parte/s.*



Agregaciones físicas: Coche / Ruedas

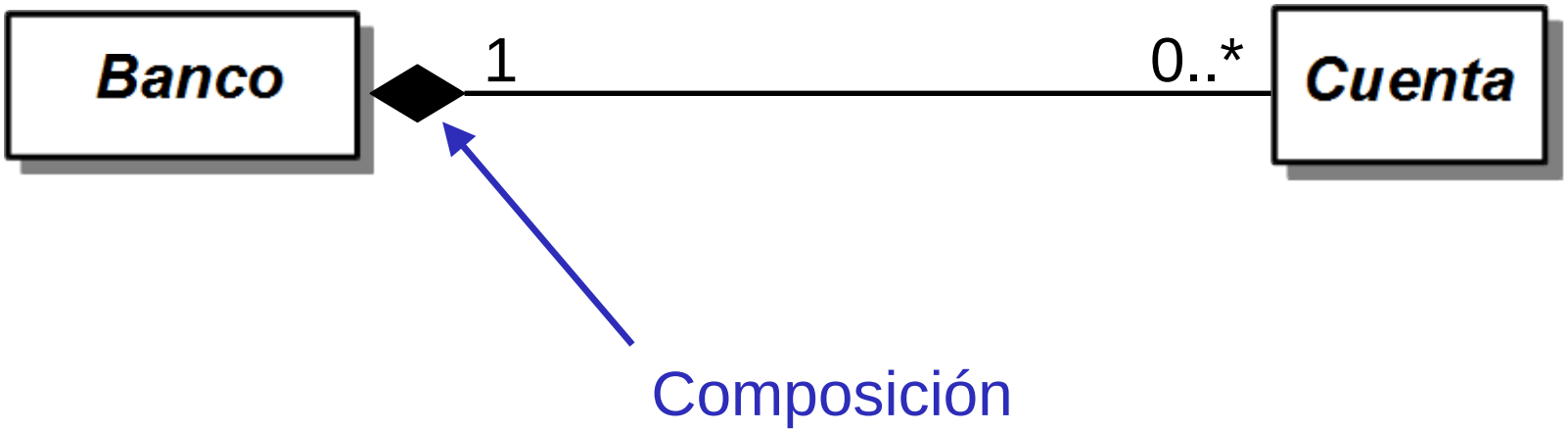
Agregaciones conceptuales: Empresa / Departamento

Adornos: Agregación

- La semántica de la agregación permite que:
  - El conjunto puede existir algunas veces independientemente de las partes, y algunas veces no.
  - Las partes pueden existir independientemente del conjunto.
  - El conjunto está en cierto sentido incompleto si falta algunas de las partes.
  - Es posible tener propiedad compartida de las partes por varios conjuntos.
- La agregación es transitiva.
- La agregación es asimétrica, es decir, un objeto nunca puede ser, directa o indirectamente parte de sí mismo.

Adornos: Composición

La **Composición** *es una forma de agregación con una fuerte relación de pertenencia y vida de las partes con el todo.*

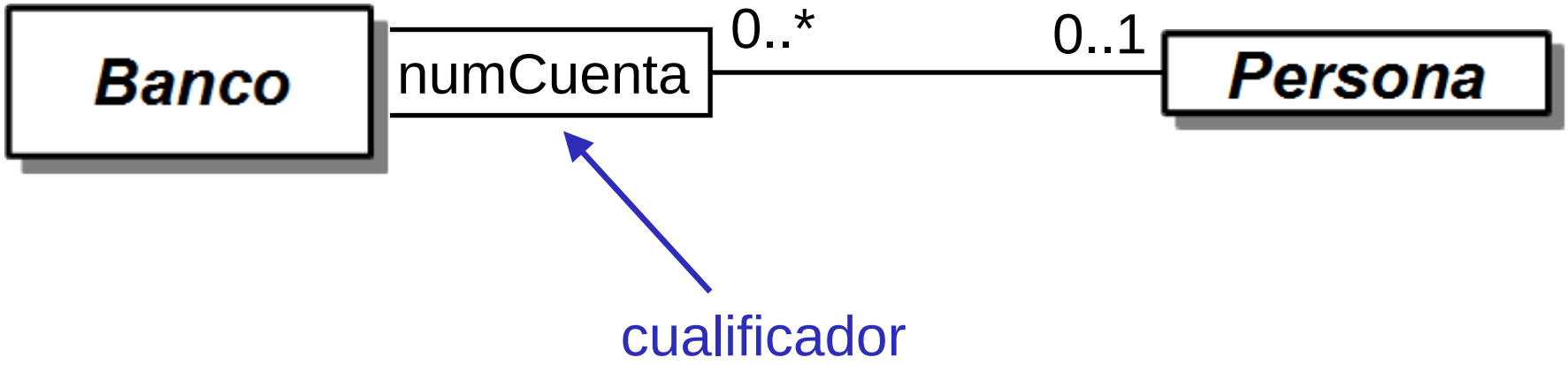


Adornos: Composición

- La semántica de la composición permite que:
  - Cada “parte” pertenece al menos a un y sólo a un “todo”.
  - Las “partes” no tienen vida independiente fuera del todo, es decir son creadas, viven y mueren con el objeto que representa el “todo”.
- La composición es transitiva.
- La composición es asimétrica.

Adornos: Cualificador

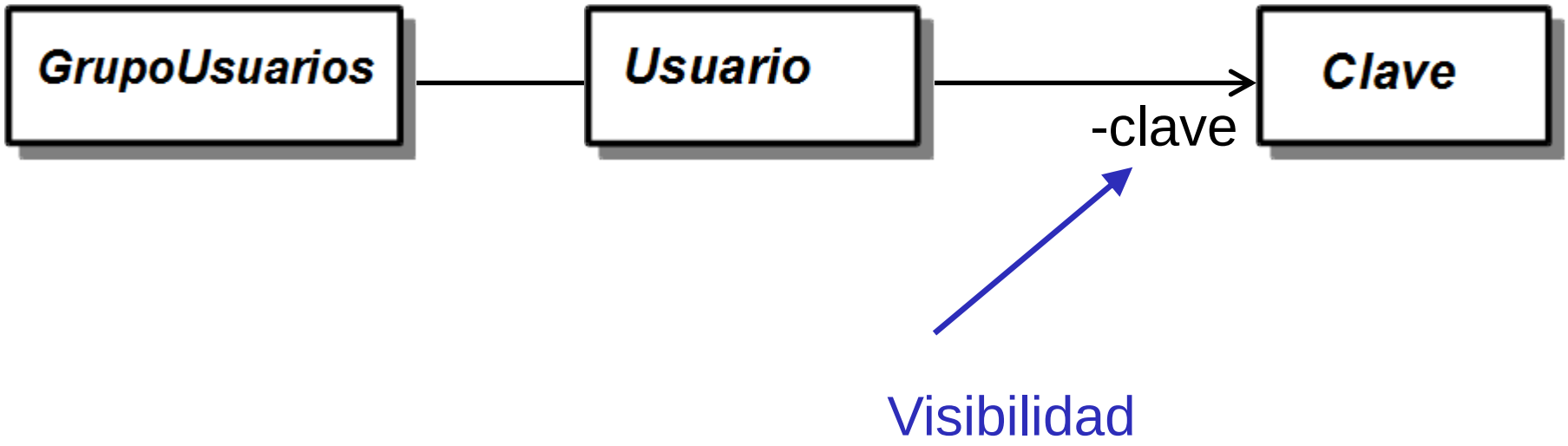
- El **cualificador** es un atributo de algunas de las clases de la asociación que pasa a ser un atributo asociado a la clase del otro extremo.
- Las asociaciones cualificadas se pueden utilizar para reducir una asociación n-a-muchos a una asociación n-a-1.



La cualificación reduce la multiplicidad del extremo opuesto al que cualifica.

Adornos: Visibilidad

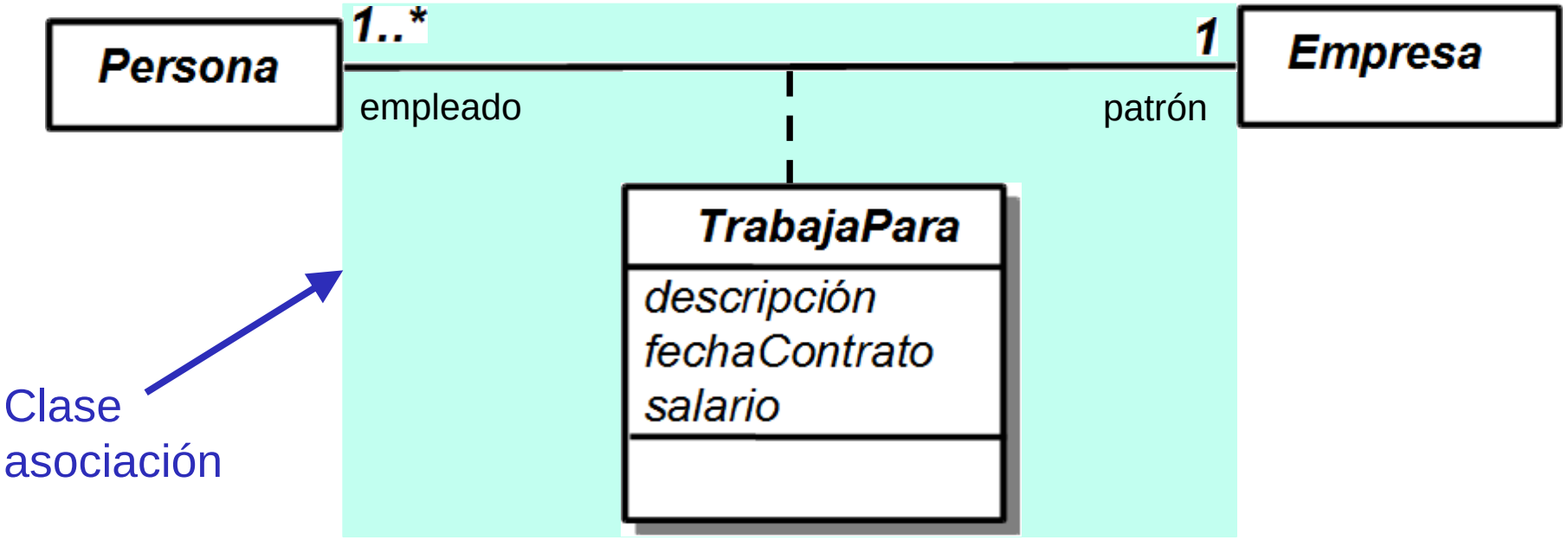
La **Visibilidad** posibilita el acceso desde los objetos de una clases a los objetos de otras clases a través de asociaciones (+,-).



Clases asociación

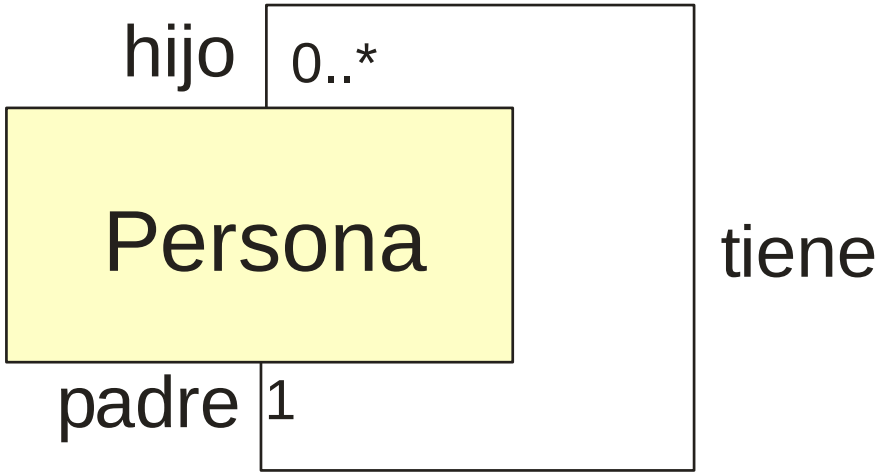
Clases asociación: Cuando una asociación presenta propiedades, hay que modelarla como una clase.

La clase asociación es la línea de asociación (incluidos los nombres de rol y multiplicidades), la línea de puntos y el cuadro de clase en el extremo de la línea de puntos.



Asociaciones reflexivas

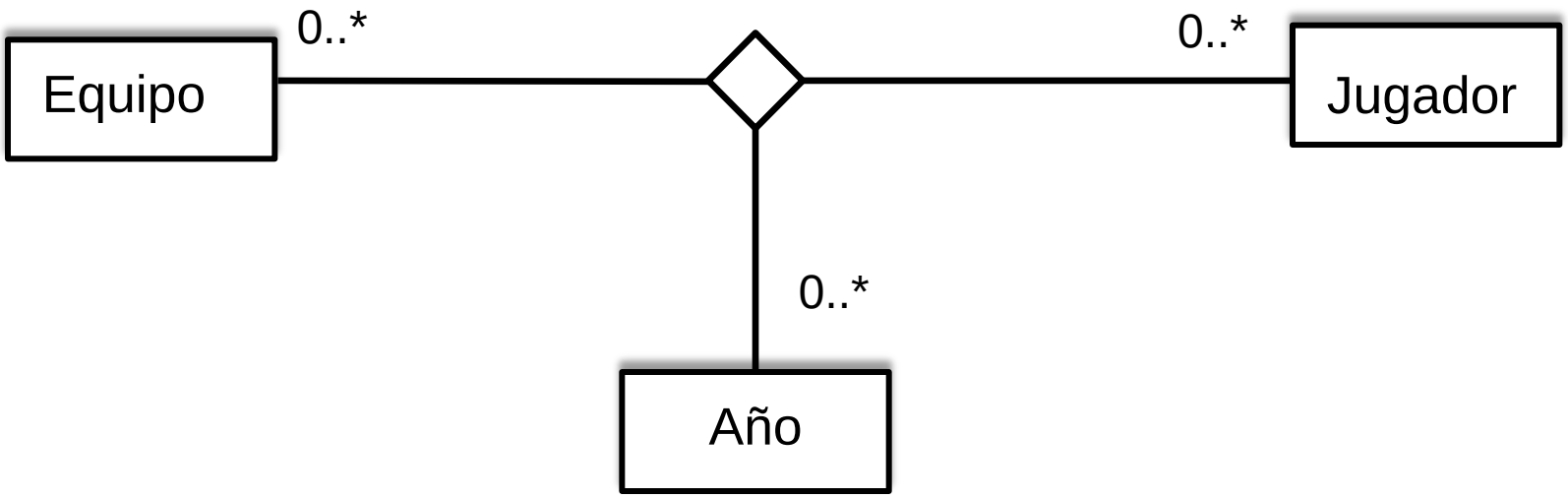
Asociaciones reflexivas: En una asociación reflexiva los objetos de una clase tienen enlaces con otros objetos de la misma clase.



Asociaciones n-arias

Asociaciones n-arias: Asociaciones que se establecen entre tres o más clases.

Cada instancia de la asociación es una n-tupla de valores, uno para cada una de las clases que componen la asociación.



Relación de Generalización

Generalización *(relación "es-un"): Es una relación entre un elemento general (padre o superclase) y un caso específico de éste (hijo o subclase)".*

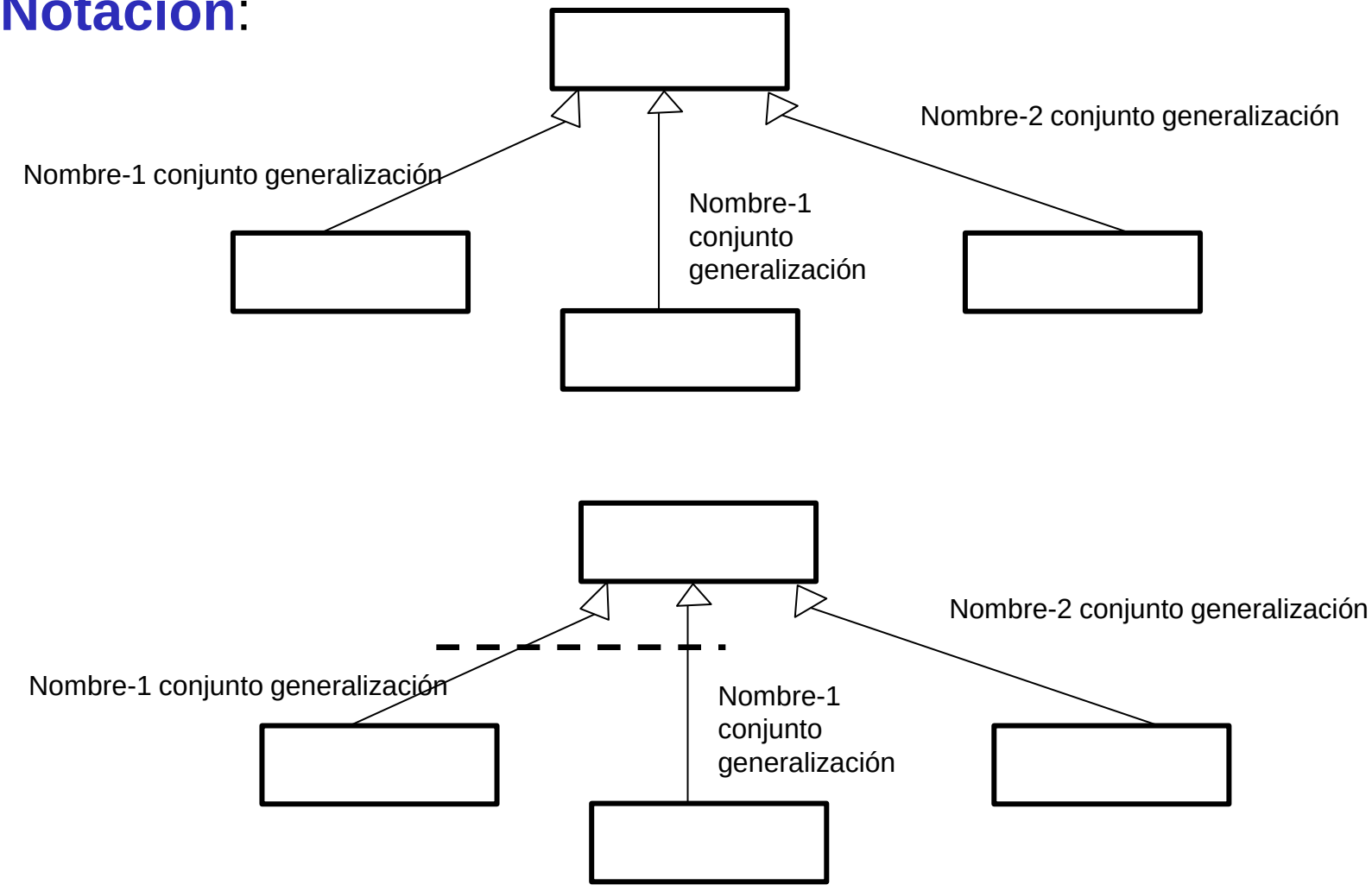
- Las subclases **heredan** las siguientes características: atributos, operaciones, relaciones y restricciones de la superclase.

- Las subclases pueden:

  - **Añadir** nuevas características.
  - **Redefinir** la implementación de las operaciones heredadas.

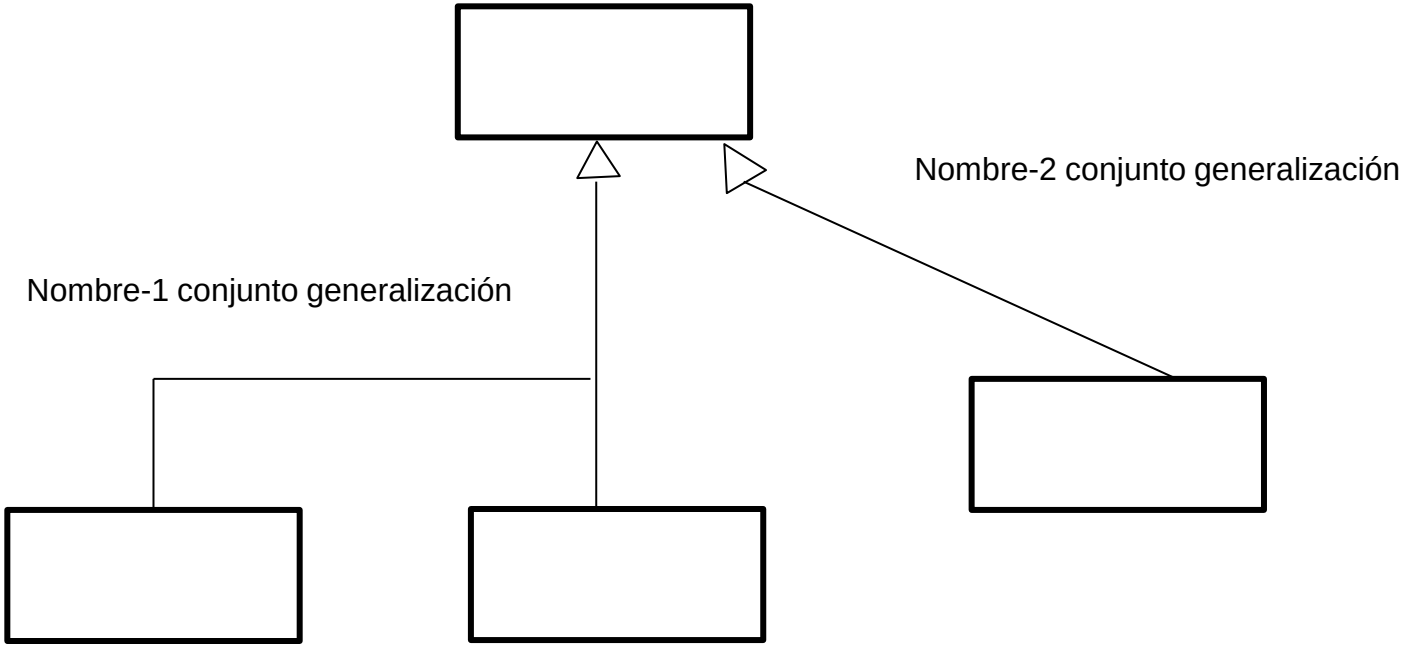
Relación de Generalización

- **Notación**:



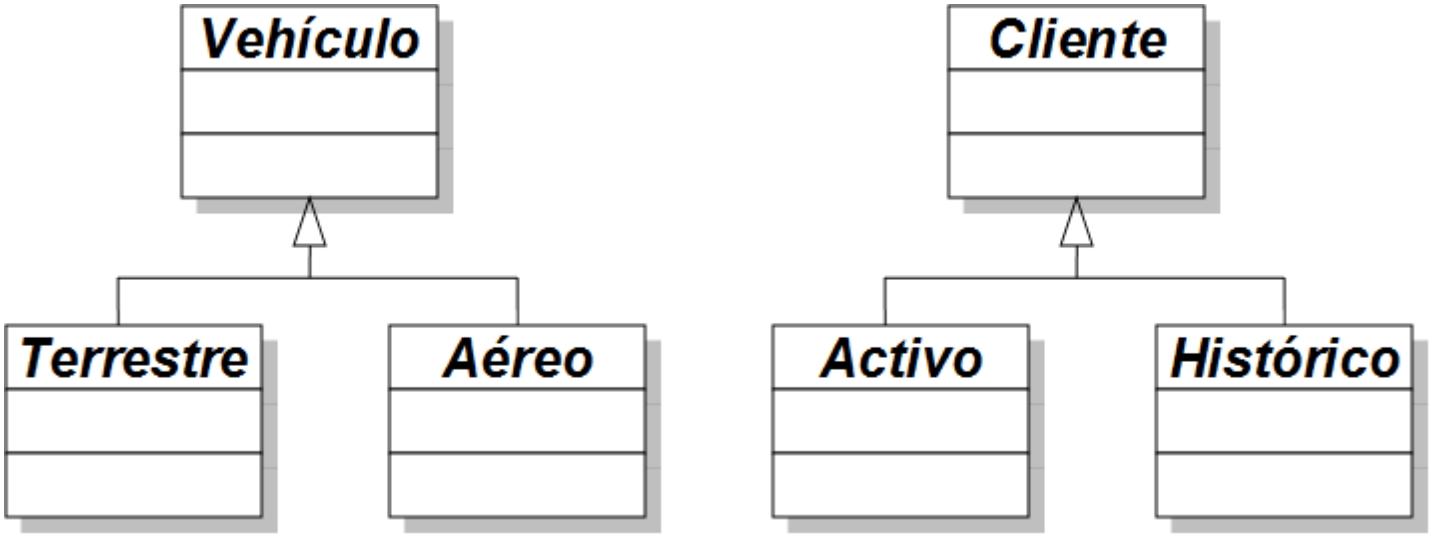
Relación de Generalización

- **Notación**:

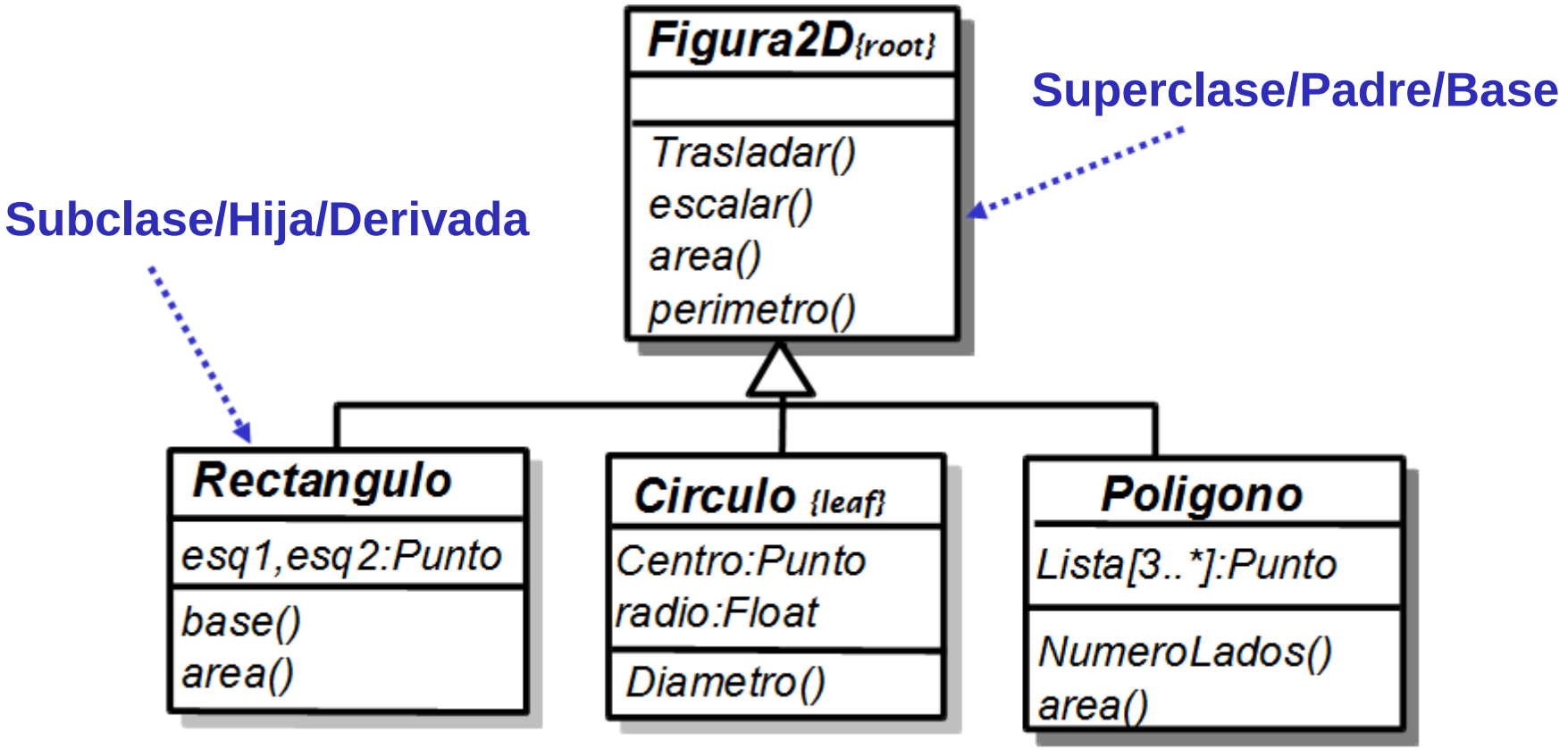


Relación de Generalización

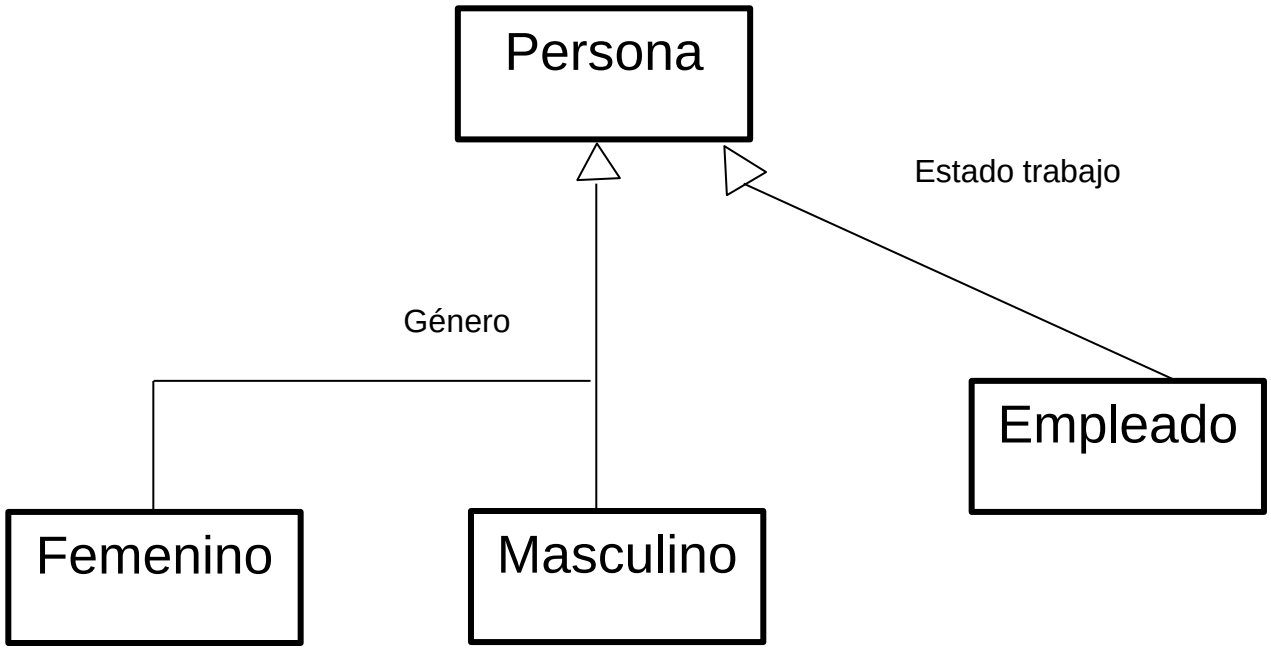
- *La* **generalización** *provoca una partición del espacio de los objetos de la superclase.*
  - **Clasificación estática**. *Espacio de los objetos.*
  - **Clasificación dinámica**. *Espacio de estados de los objetos.*



Relación de Generalización



Relación de Generalización



Relación de Generalización

Los conjuntos de generalización pueden tener restricciones aplicadas:

- **{complete}**: Las subclases en el conjunto de generalización abarcan todas las posibilidades.
- **{incomplete}**: Puede haber subclases que no estén en el conjunto de generalización.
- **{disjoint}**: Un objeto puede ser una instancia de uno y solamente uno de los miembros del conjunto de generalización.
- **{overlapping}**: Un objeto puede ser una instancia de más de uno de los miembros del conjunto de generalización.

Estas restricciones se combinan como:

{incomplete, disjoint} **{complete, disjoint}**

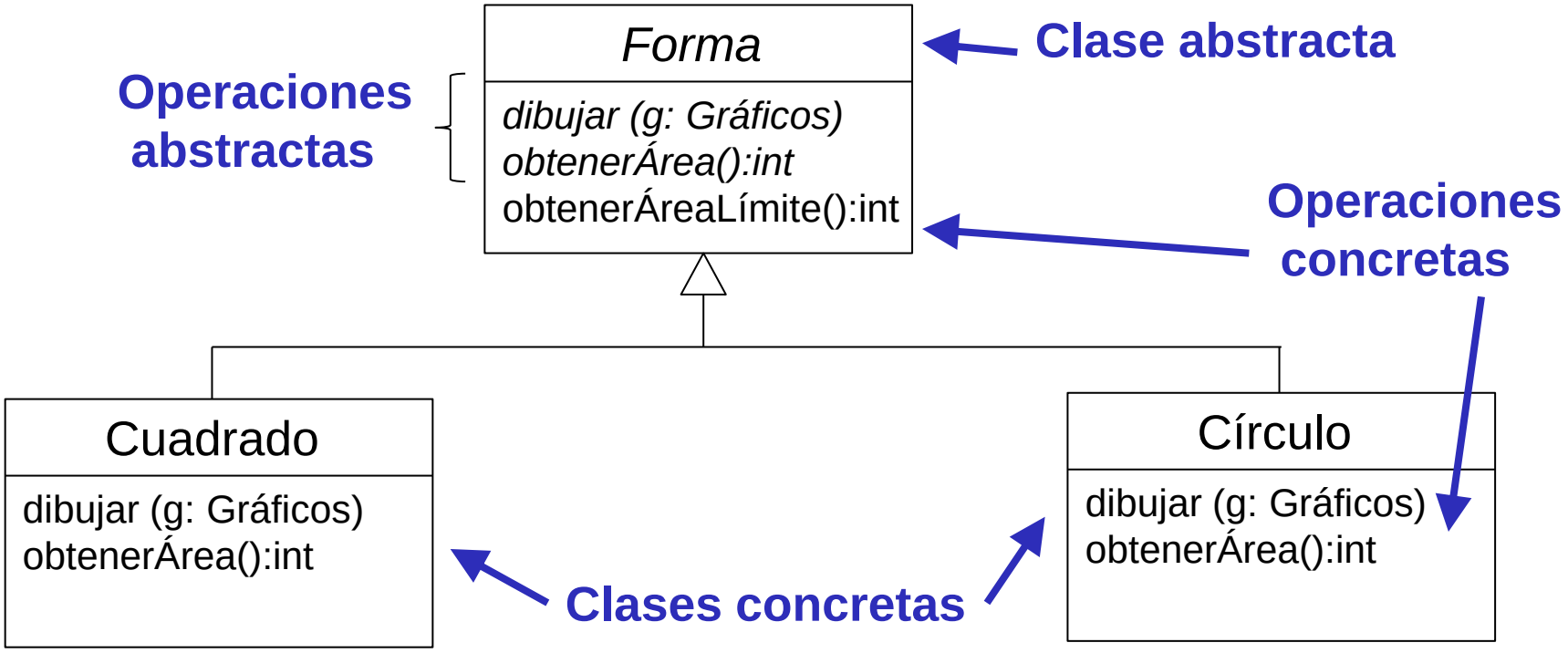
(predeterminado)

{incomplete, overlapping} **{complete, overlapping}**

Clase Abstracta

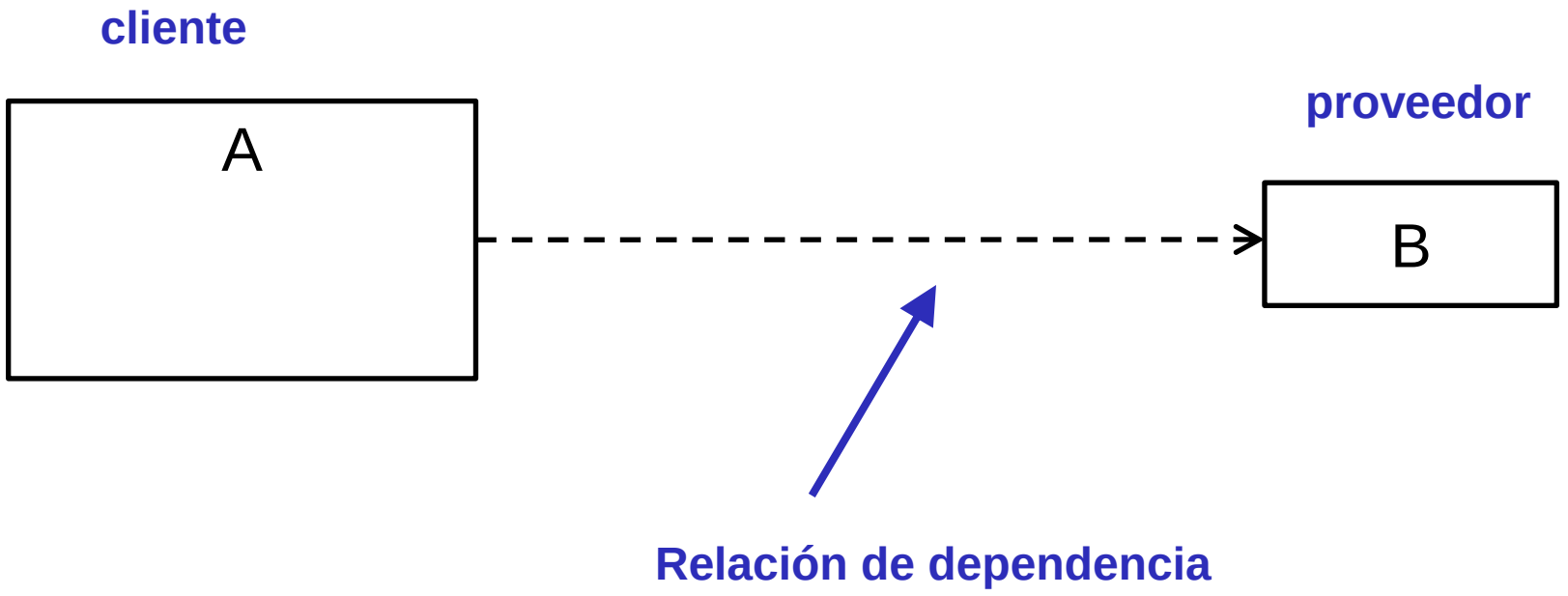
Clase abstracta: Es una clase no instanciable que posee al menos un método abstracto (sin implementación, es decir, aún no definidos), que se implementa a través de especializaciones en subclases.

– Se especifican escribiendo el nombre en cursiva.



Relación de Dependencia

Relación de dependencia: Indica una relación semántica de uso entre dos o más elementos del modelo donde un cambio en un elemento (el proveedor) puede causar cambios en el elemento dependiente.

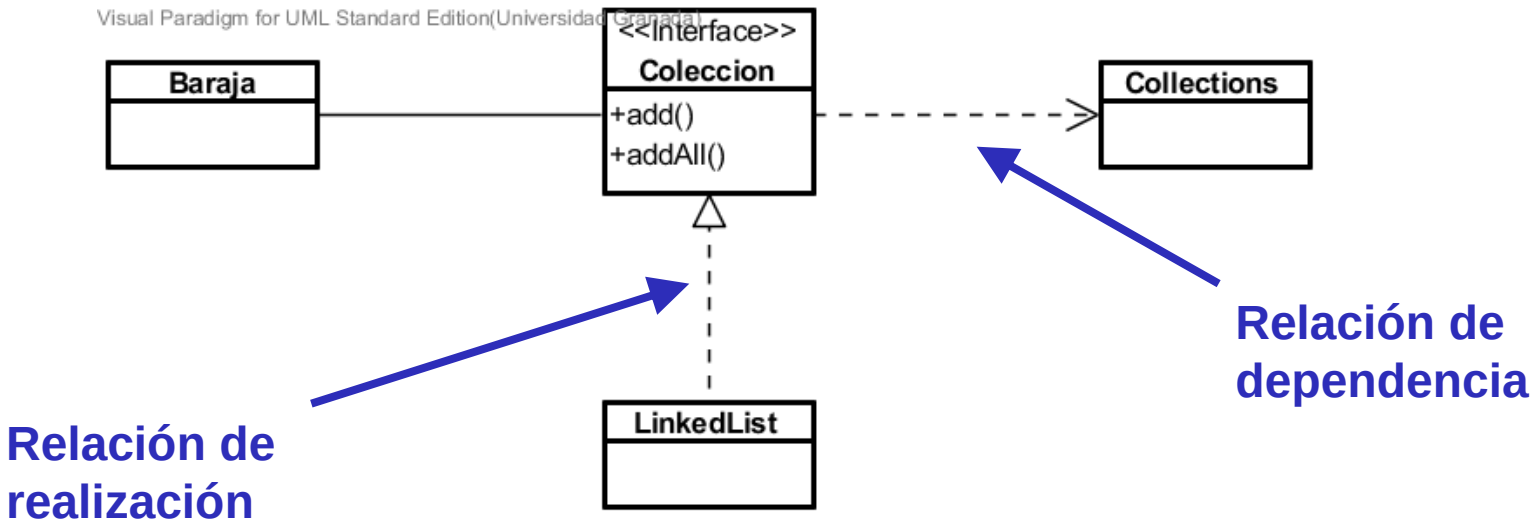


Las dependencias no necesitan tener un nombre.

Relación de Realización

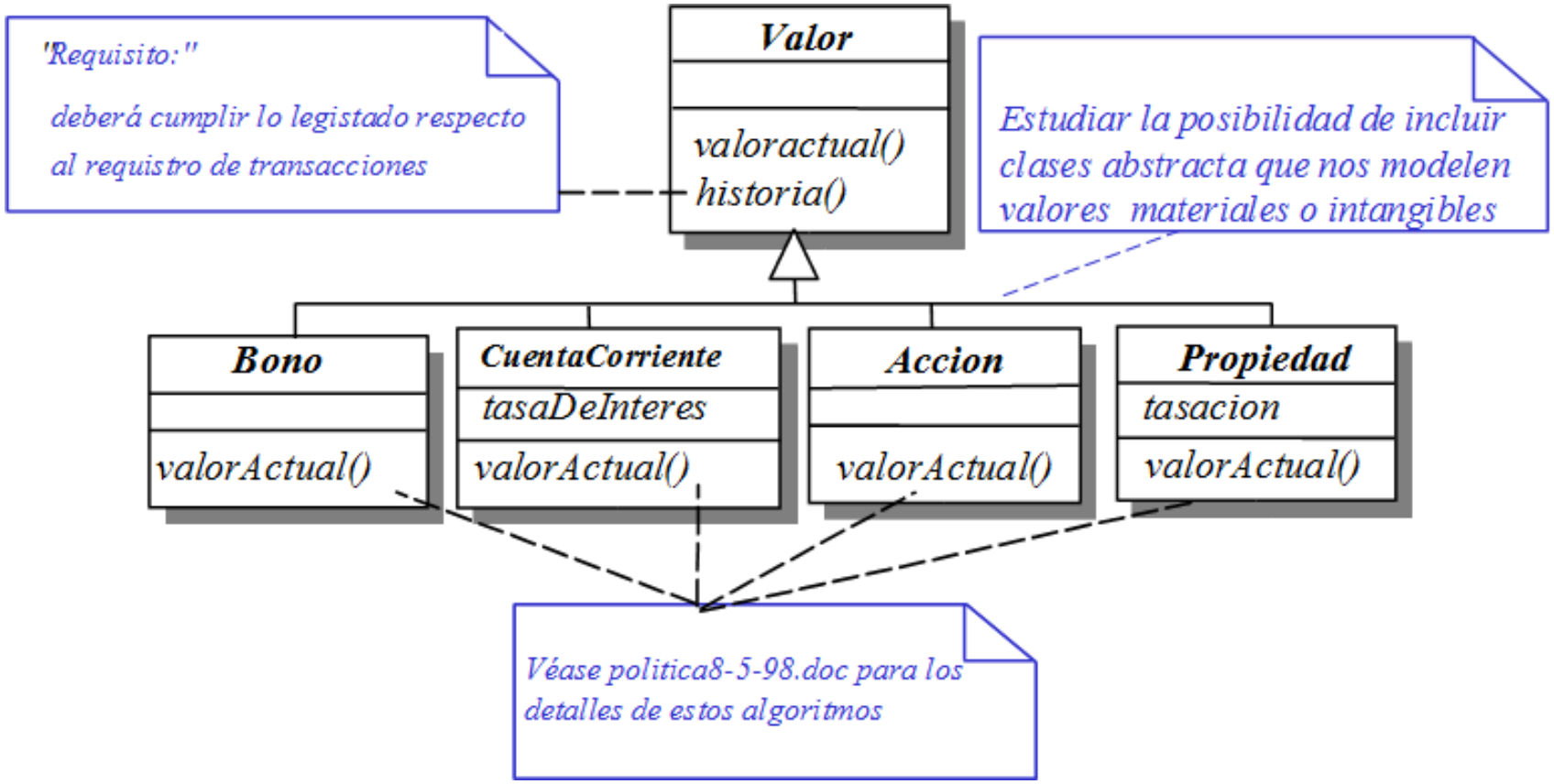
Relación de realización: Se establece entre dos elementos cuando uno de ellos especifica un contrato y el otro garantiza que se cumple.

Por ejemplo: Una interfaz presenta una **relación de realización** con la/s clase/s que la implementa/n.



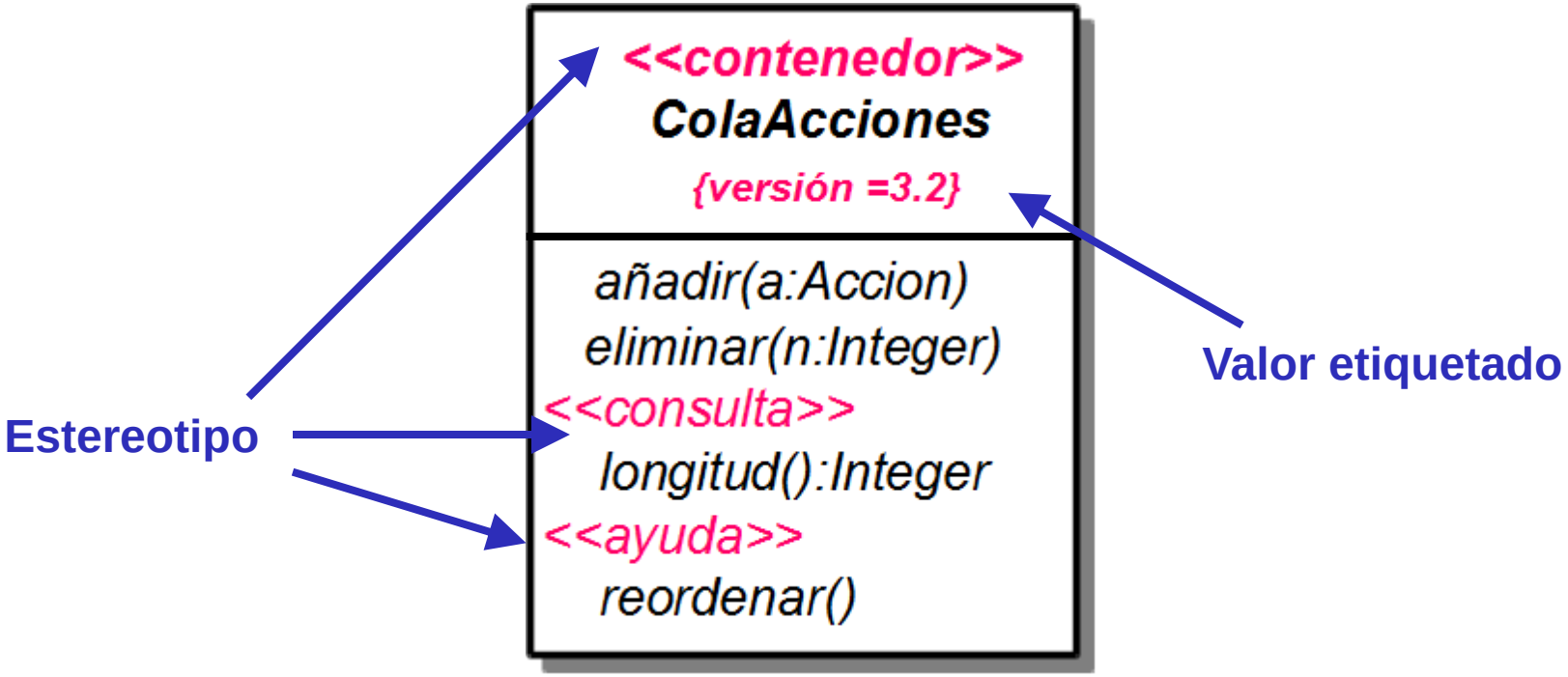
Notas

Nota: Es un símbolo gráfico o textual para representar comentarios asociados a uno o varios componentes.



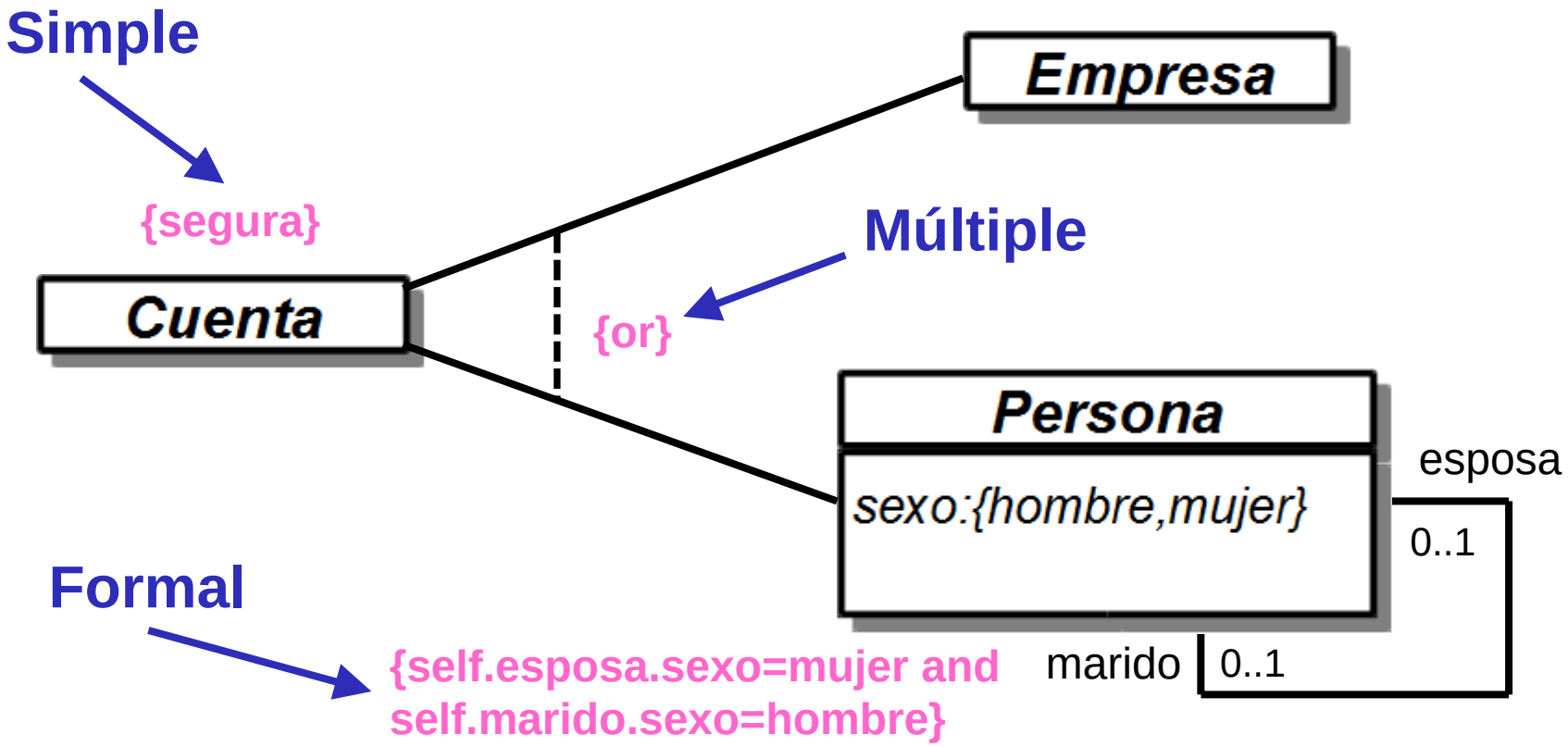
Mecanismos de extensión

- **Estereotipos**: Añaden nuevos elementos de construcción. Se representan entre << >>.
- **Valores etiquetados**: Añaden nuevas propiedades sobre los elementos del lenguajes. Se representan entre { }.

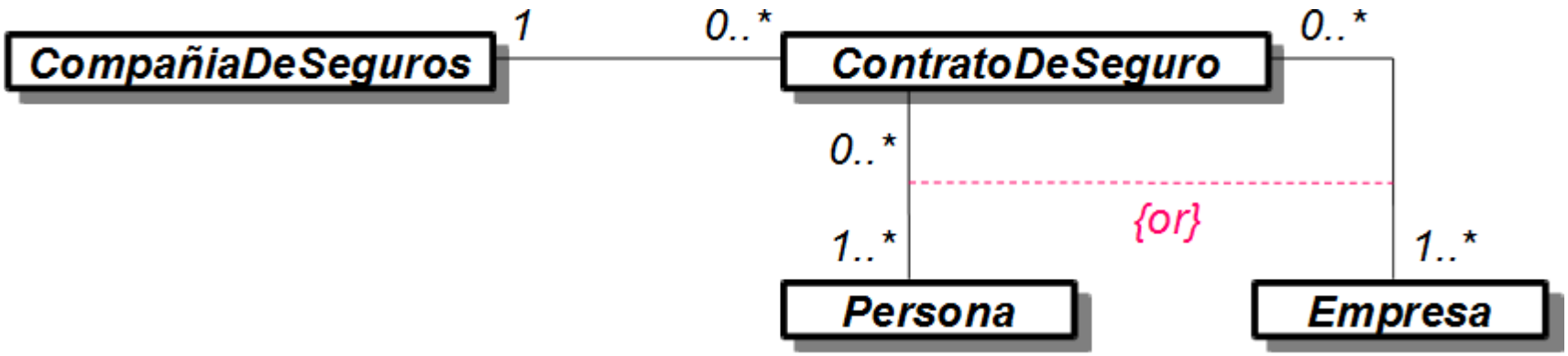


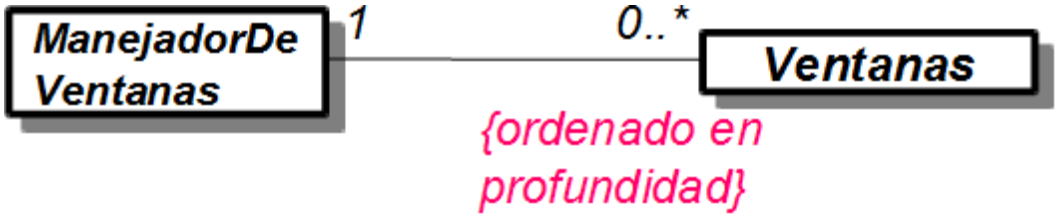
Mecanismos de extensión

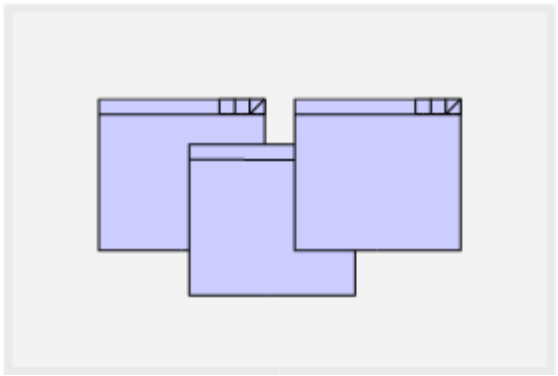
- **Restricciones**: Son extensiones de la semántica de los elementos del lenguaje, añadiendo nuevas reglas o modificando las existentes.



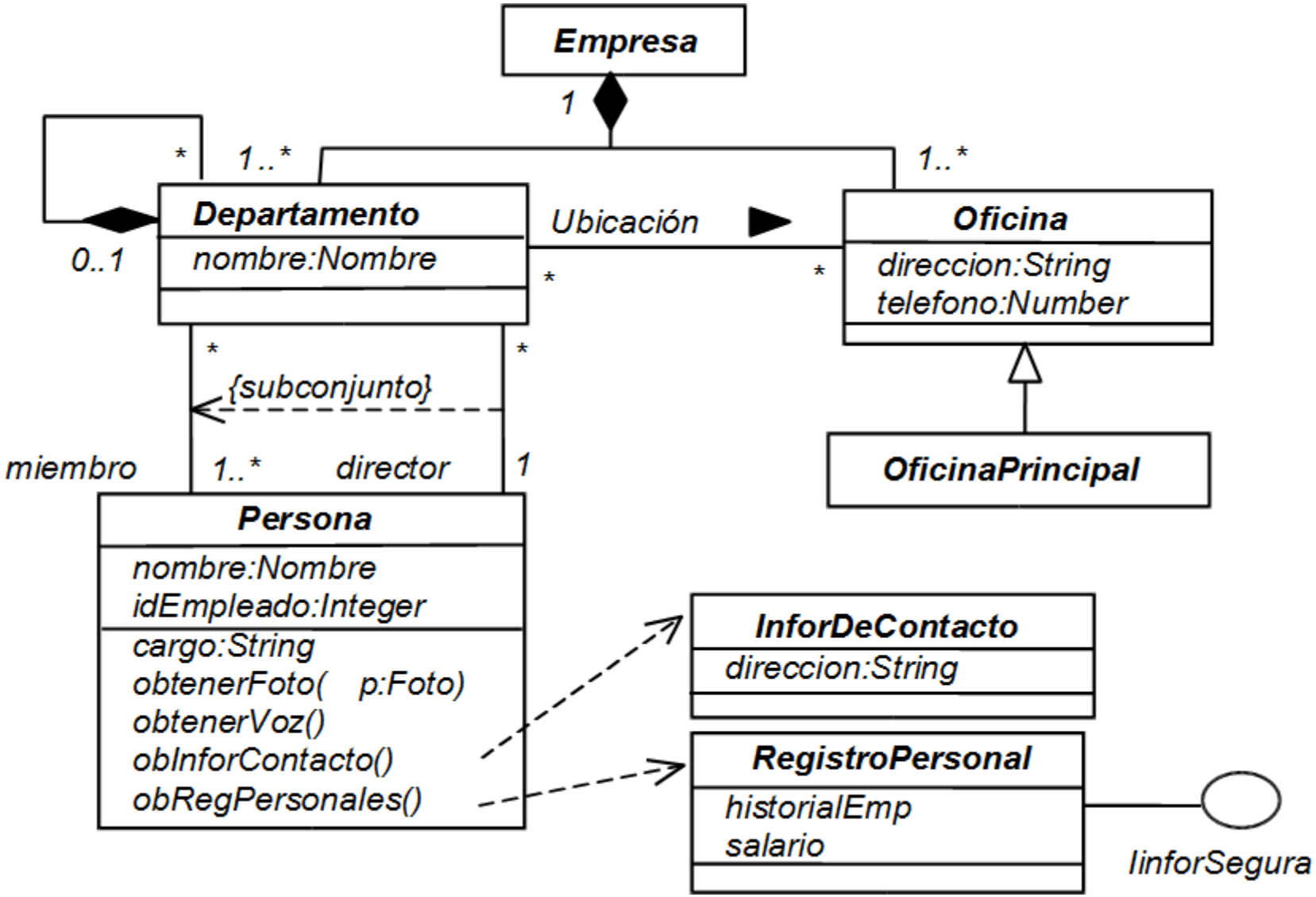
Ejemplos de restricciones







Ejemplo: Diagrama de clases



Ejemplo: Diagrama de clases