Modelado Estático Básico

Universidad
Carlos III de Madrid

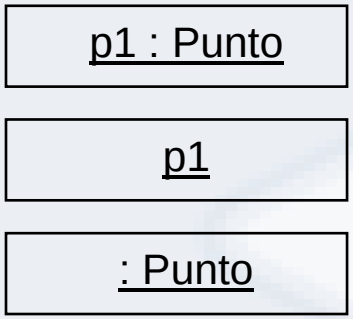
Objetos y Clases (I)

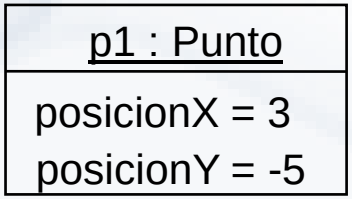
- Dos niveles de abstracción:
  - **Objeto**: representación de una entidad concreta con identidad, estado y comportamiento (no siempre entidades físicas tangibles).
  - **Clase**: especificación de un conjunto de entidades con estructura y comportamiento comunes.

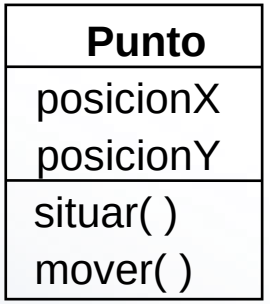
- Ejemplos de clases:
  - Objetos físicos: avión, persona, libro...
  - Objetos lógicos: cuenta corriente, asignatura, número complejo.
  - Objetos históricos: asiento bancario, reserva de habitación.
  - Objetos tipo: producto a la venta, ingrediente de una receta.

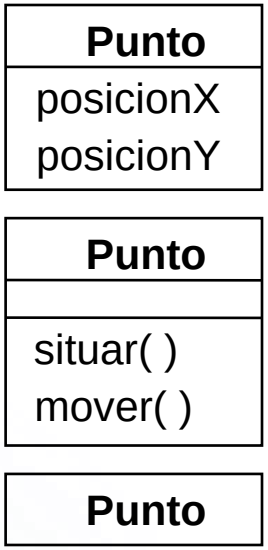
Objetos y Clases (y II)

- Supresión opcional de compartimentos

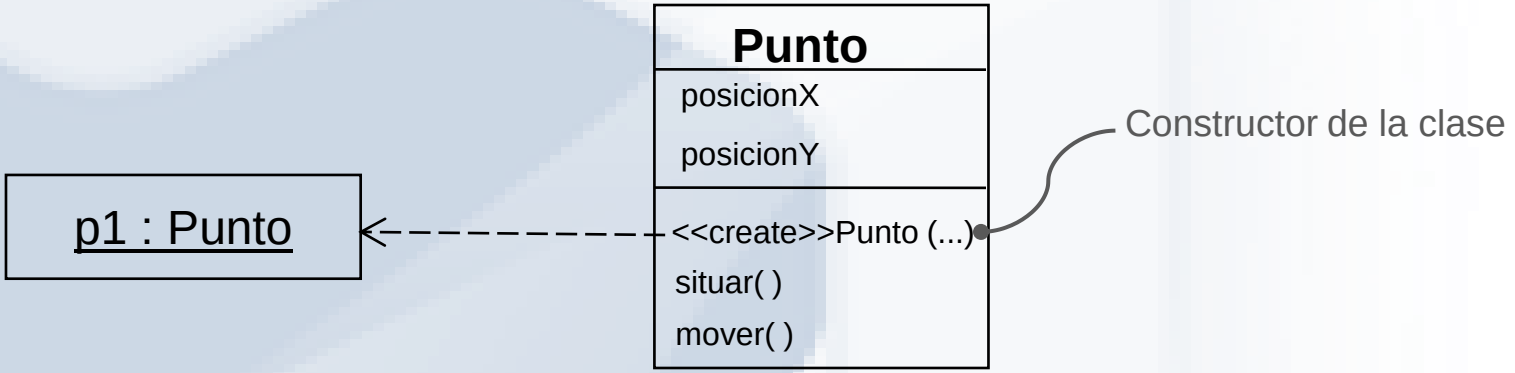








- Relación entre un objeto y su constructor:



Atributos y Operaciones (I)

- **Atributo**: es una propiedad compartida por los objetos de una clase.
  - Cada atributo tiene un **valor** (probablemente diferente) para cada objeto.
  - Atributo **derivado**: propiedad redundante que puede ser calculada a partir de otras.
    - ***/*area** (= base * altura).
    - Pueden implementarse como operaciones.
- **Operación**: es una función o transformación que se puede aplicar a los objetos de una clase.
  - Pueden ser **invocadas** por otros objetos, o por el mismo objeto.
  - **Método**: especificación procedimental (implementación) de una operación.
- (...) : para indicar que la lista de atributos/operaciones no está completa.

Atributos y Operaciones (y II)

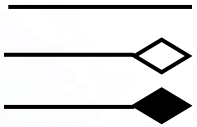
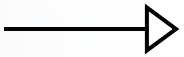
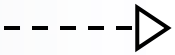
- **Notación** (pueden suprimirse todos los elementos excepto el nombre):
- [<visibilidad>] [‘/’] **<nombre-atributo>** [‘:’ <tipo> [‘[‘<multiplicidad>‘]’] [‘=‘ <valor-inicial>]]
  - saldo : Moneda = 0
  - telefonoOficina: String[0..2]
- [visibilidad] **<nombre-operación>** ‘(‘ [<parámetro> [‘,’ <parámetro>]*]‘)’ [‘:’ <tipo-retorno>]
  donde, <parámetro> ::= [‘in’|’out|’inout’] <nombre-parámetro> ‘:’ <tipo> [‘[‘<multiplicidad>’]’] [‘=‘ <valor-inicial>]] (valor por omisión: ‘in’)
  - obtenerSaldo ( ) : Moneda
  - marcar (numero : Telefono, reintentos : Integer)
  - +crearVentana(posicion:Coordenadas,contenedor:Contenedor[0..1]):Ventana
  - -ocultar()

Visibilidad

- Niveles de visibilidad (diferentes en cada lenguaje):
  - public: visible para todas las clases (opción por defecto para **operaciones**).
  - package: visible para todas las clases que estén en el mismo paquete.
  - protected: visible para las subclases.
  - private: visible sólo para la clase (opción por defecto para **atributos**).

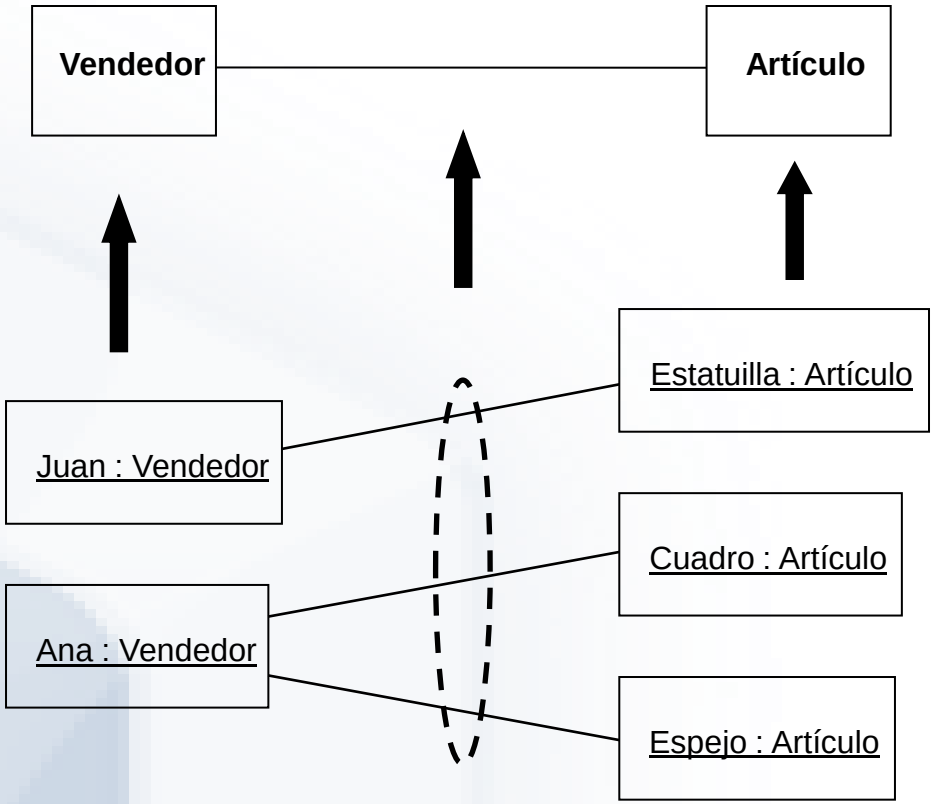
<table>
<tr><th colspan="2">UML</th><th>Java</th><th colspan="2">VB.NET</th></tr>
<tr><td colspan="2">Public</td><td>Public</td><td colspan="2">Public</td></tr>
<tr><td rowspan="2">Package</td><td rowspan="2">Protected</td><td>Protected</td><td colspan="2">Protected-Friend</td></tr>
<tr><td>friendly</td><td>Friend</td><td>Protected</td></tr>
<tr><td colspan="2">Private</td><td>Private</td><td colspan="2">Private</td></tr>
</table>

Tipos de Relaciones

Dependencia	El elemento dependiente requiere la presencia del elemento independiente para su especificación o implementación. Los cambios en el elemento independiente pueden afectar al elemento dependiente.	
Asociación	Especificación de un conjunto de conexiones entre instancias. Representan la estructura y posibilidades de comunicación del sistema.	
Generalización	Relación entre un elemento general y un elemento específico. El elemento específico puede añadir información y debe ser consistente con el elemento general.	
Realización	Relación donde un elemento realiza o implementa la especificación dada por otro elemento.	

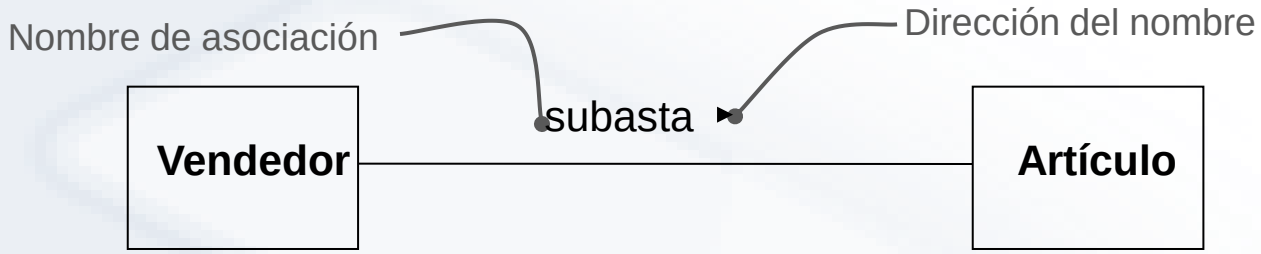
Enlaces y Asociaciones

- **Asociación**: especificación de un conjunto de enlaces entre objetos de las clases asociadas.
  - Representa la **estructura** y el **comportamiento** del sistema.
- **Enlace**: conexión entre objetos.
  - Determina una tupla de objetos. Instancia de una asociación.
  - Estado de los objetos enlazados.
  - Estado del sistema.
- **hecho** + posibilidad de **comunicación**

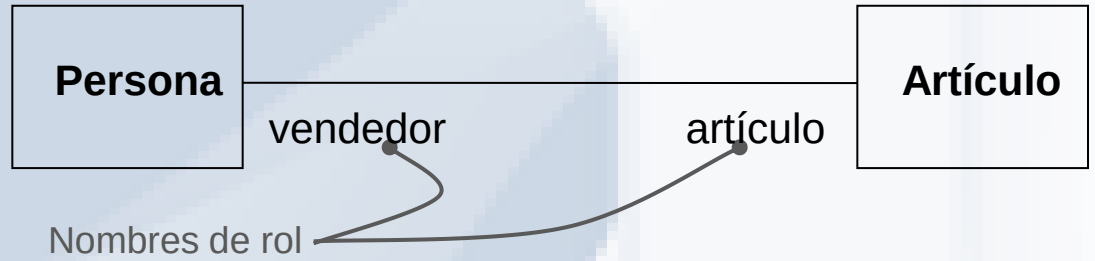


Nombre de Asociación y Nombre de Rol

- Los nombres de asociación se pueden repetir en un modelo, excepto para asociaciones entre las dos mismas clases.

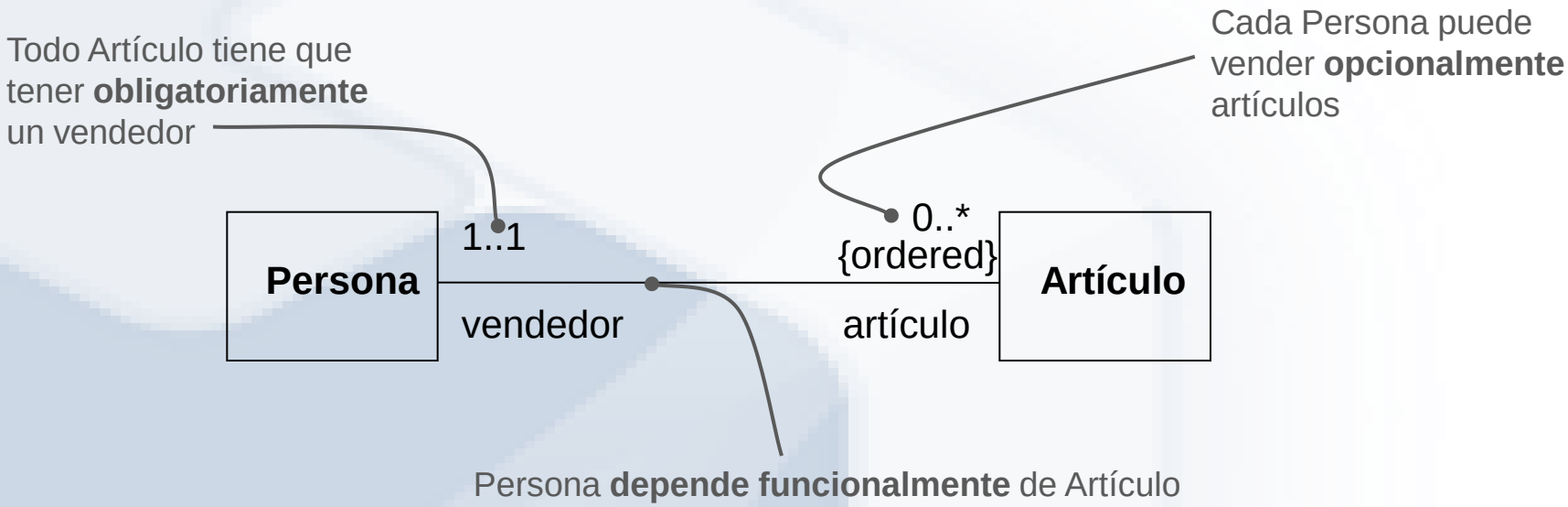


- Los nombres de rol se pueden repetir en asociaciones distintas, y pueden ser iguales que los nombres de las clases asociadas.



Multiplicidad de la Asociación (I)

- En una **asociación binaria**, la multiplicidad de un **extremo de asociación** especifica el número de instancias **destino** que pueden estar enlazadas con una única instancia **origen** a través de la asociación.



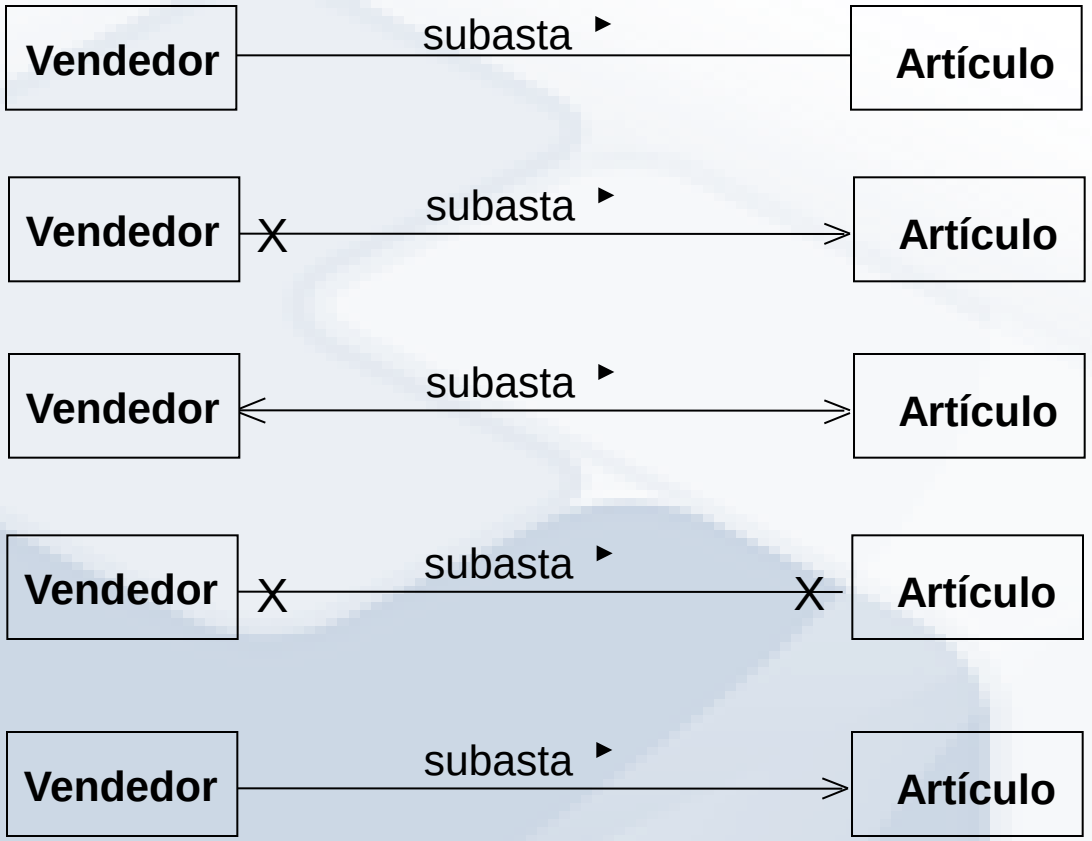
Multiplicidad de la Asociación (y II)

- **Valores típicos**:
  - 0..1 cero o uno
  - 1..1 uno y sólo uno (abreviado como “1”)
  - 0..* desde cero hasta “muchos” (abreviado como “*”)
  - 1..* desde uno hasta “muchos”
- **Otros valores**:
  - Rangos enteros: (2..*), (0..3), etc.
  - Lista de rangos separados por comas: (1, 3, 5..10, 20..*), (0, 2, 4, 8), etc.
- **Si no se indica multiplicidad se considera 1**.
- **Restricciones (entre llaves):**
  - ordered/unordered: las instancias van ordenadas / o no (valor por omisión: {unordered}).
  - unique/nonunique: los valores de las instancias deben ser únicos / o no (valor por omisión: ‘unique’). Se pondrá ‘nonunique’ para permitir tener varios enlaces que asocien el mismo conjunto de instancias.

Navegabilidad de la Asociación (I)

- La navegabilidad de una **asociación binaria** especifica la capacidad que tiene una instancia de la clase **origen** de acceder a las instancias de la clase **destino** por medio de las **instancias de la asociación** que las conectan.
- **Acceder** = nombrar, designar o referenciar el objeto para...
  - leer o modificar el valor de un **atributo** del objeto (desaconsejable),
  - invocar una operación del objeto (enviarle un mensaje),
  - usar el objeto como **parámetro** o **valor de retorno** en un mensaje,
  - modificar (asignar o borrar) el **enlace** con el objeto.
- No confundir:
  - **Dirección del nombre** de la asociación: asimetría lingüística.
  - **Navegabilidad** o direccionalidad de la asociación: asimetría comunicativa.

Navegabilidad de la Asociación (y II)



Navegabilidad no especificada/asociación bidireccional

Asociación unidireccional

Asociación bidireccional

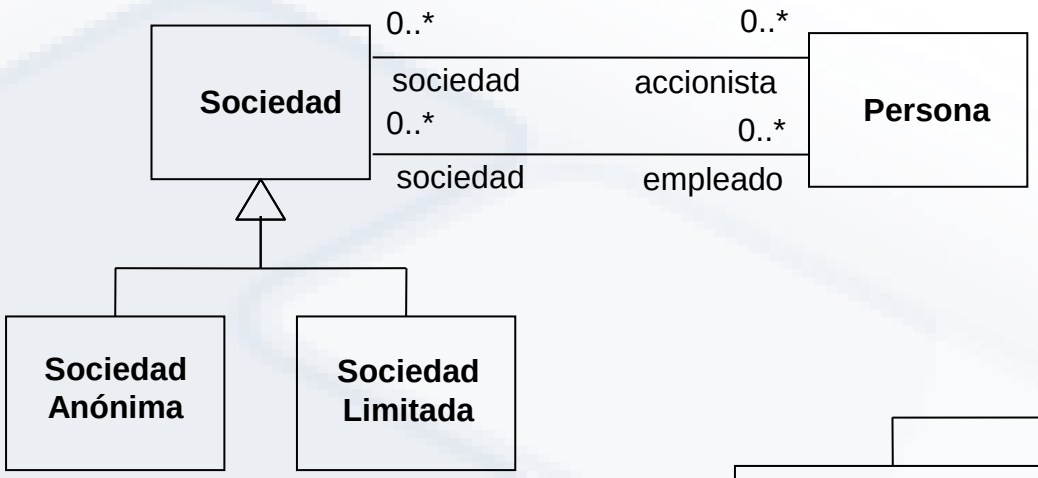
Asociación sin navegabilidad

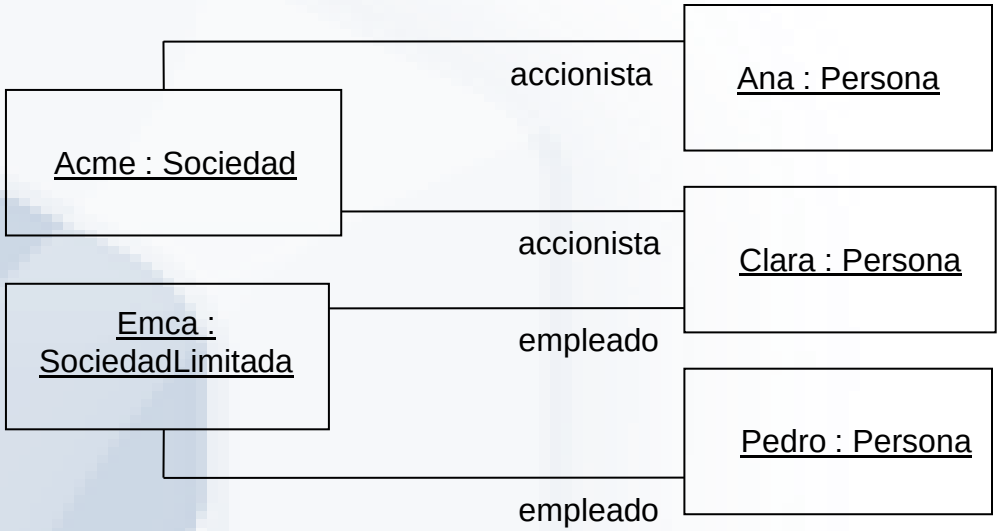
Asociación con navegabilidad en un sentido, y sin especificar en el otro

Diagramas de Clases y de Objetos (I)

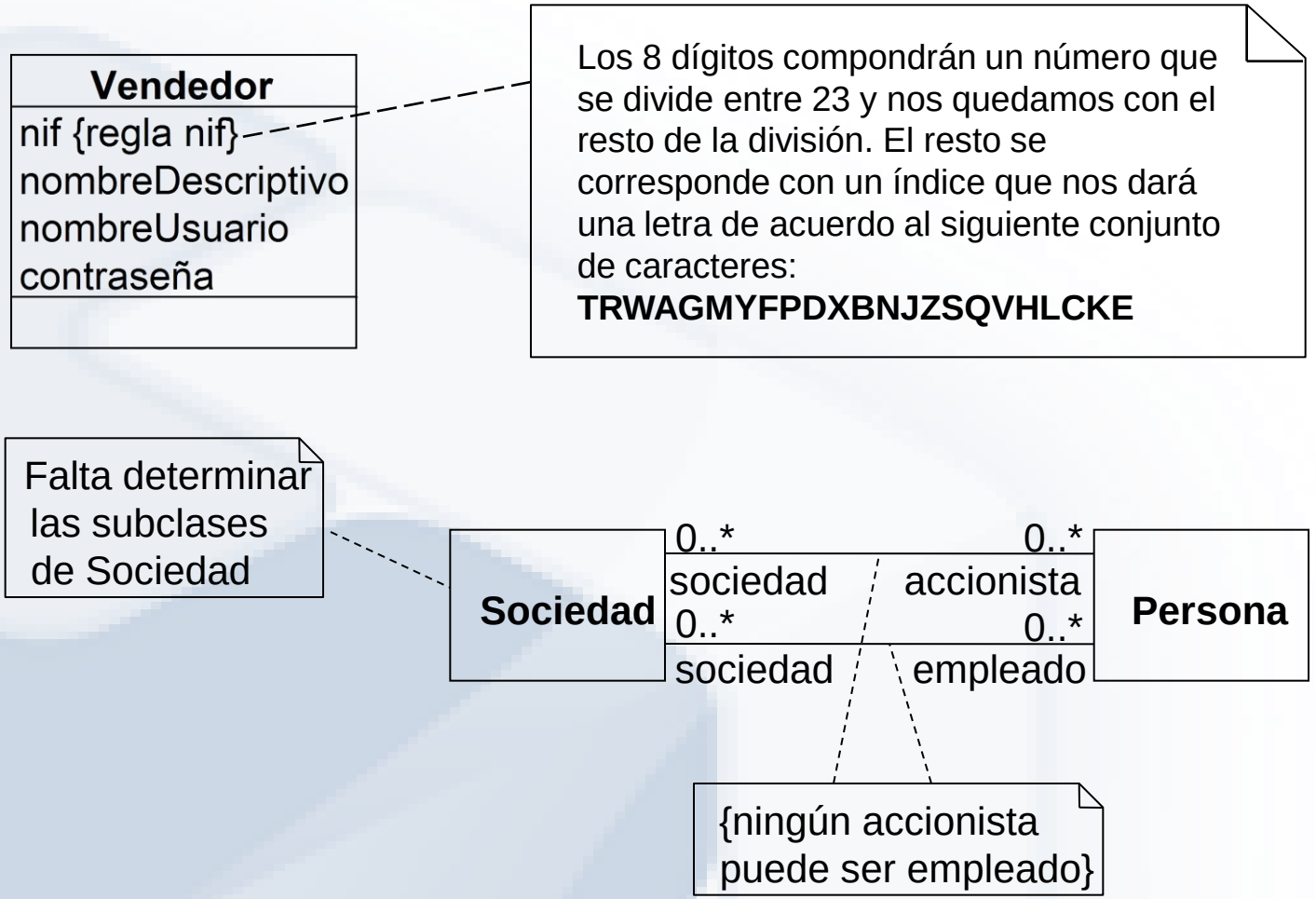
- Diagrama de clases
  - Captura y **especifica** el vocabulario del sistema:
    - **Elementos**: clases, atributos, operaciones...
    - **Relaciones**: asociaciones, generalizaciones...
    - **Estructura** del sistema, fundamento de su **comportamiento**
  - Sugerencias para **mejorar la comunicación**:
    - Nombres adecuados: clases, atributos, operaciones, asociaciones, roles.
    - Distribución espacial de los elementos.
    - Evitar cruces de líneas.
    - Distinto nivel de detalle según el propósito y nivel de abstracción.
- Diagrama de objetos
  - **Ilustra** la estructura del sistema mediante situaciones particulares.
  - "Fotografía" del sistema: objetos, valores de atributos; enlaces.
  - Las instancias deben **conformarse** a sus especificaciones.
    - Objetos, enlaces → clases, asociaciones.
    - Las especificaciones pueden estar representadas en **distintos diagramas**.

Diagramas de Clases y de Objetos (y II)



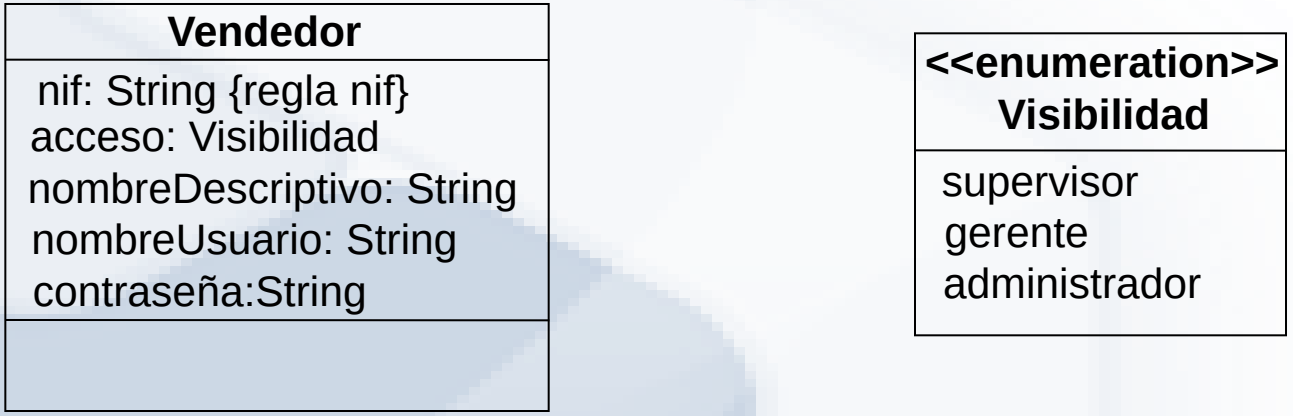


Restricciones y Notas



Tipos Enumerados

- Un tipo enumerado es un tipo de datos cuyos valores se enumeran **en el modelo** como literales.



Clases de Análisis y Clases de Diseño

- Análisis = modelo conceptual
  - Clases, atributos y operaciones corresponden a **conceptos del dominio**.
  - No es fácil determinar las **operaciones** antes de realizar el modelo dinámico.
  - Es habitual usar una **notación simplificada** al máximo.
- Diseño = modelo del software
  - Clases, atributos y operaciones corresponden a **fragmentos de código**.
  - Nuevos artefactos que dependen del **lenguaje** y la **plataforma** de implementación.
  - **Clases**:
    - Una clase de análisis puede ser implementada por varias clases de diseño.
    - Ejemplo: Producto = VentanaProducto + DatosProducto
  - **Atributos**:
    - Unicidad (atributo identificador).
    - Modificabilidad de los valores.
    - Mecanismos de almacenamiento persistente.
  - **Operaciones**:
    - Se define la implementación de las operaciones mediante métodos.