

---

## TEMA 4

# UML (UNIFIED MODELING LANGUAGE)

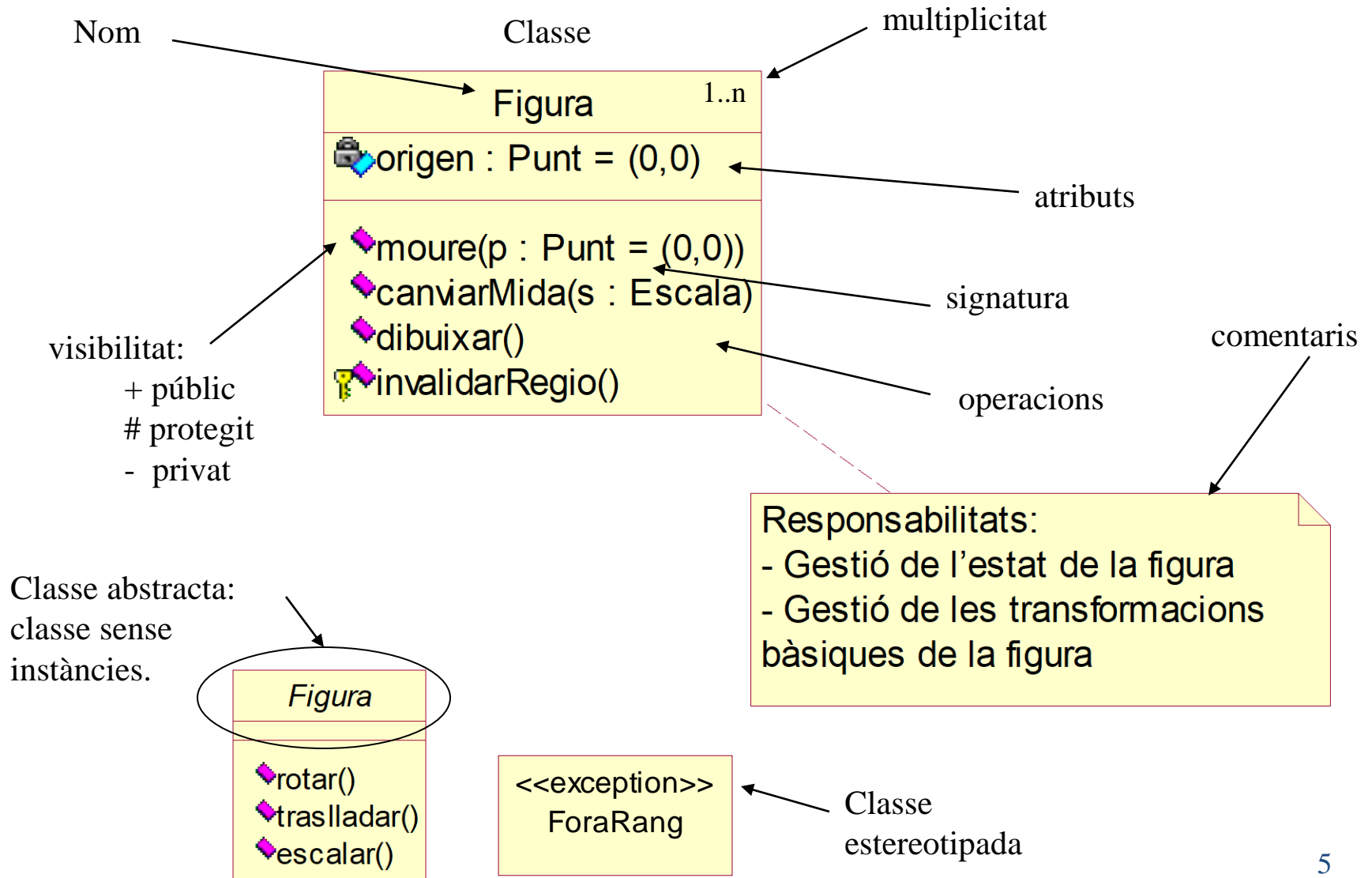
1. Introducció,
- 2. Diagrames de Classes (\*)**
3. Diagrames de Casos d'Us
4. Diagrames de Seqüència
5. Diagrames d'Activitats
6. Diagrames d'Estats
7. Altres diagrames

---

# Diagrames de Classes

- *NOTA: Els diagrames de Classes ja els coneixeu. Per tant, aquest bloc es presenta com a material ja donat. Així i tot, cal assegurar-se de que **tot el contingut** inclòs el teniu assimilat.*

- Es descriuen els tipus d'objectes d'un sistema i les **relacions estàtiques** que hi ha entre ells.
- S'expressa mitjançant els **diagrames de classe**.
- Normalment contenen:
  - Classes
  - Interfícies
  - Relacions de dependència, realització, generalització i associació (agregació, composició)
- També poden incloure paquets i col·laboracions.



## Visibilitat de les classes, atributs o operacions

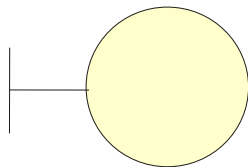
- **public:** els membres d'una classe són accessibles per tots els clients.
- **protegit:** els membres d'una classe són accessibles només per les subclasses, amigues i la mateixa classe.
- **privat:** els membres d'una classe són accessibles només per les classes amigues i la mateixa classe.
- **implementació:** la classe és accessible per la implementació del paquet que conté la classe.

Notació →

Visibilitat	
	public
 	privat
 	protegit
 	implementacio
<hr/>	
	public()
 	privat()
 	protegit()
 	implementació()

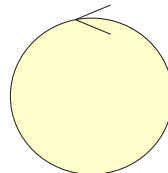
Les classes del model d'anàlisi són classes estereotipades que representen un model conceptual inicial per als elements del sistema que tenen responsabilitats i comportament. Hi ha tres tipus de classes d'anàlisi:

- **Comunicació.** Gestionen la interacció entre el sistema i el seu entorn. S'utilitzen per modelar les interfícies del sistema.
- **Control.** Coordinen els events necessaris per realitzar el comportament especificat en un cas d'ús.
- **Entitat.** Modelen informació i el seu comportament. Representen entitats del món real o entitats internes necessaries per executar les tasques del sistema. Són independents de com l'entorn es comunica amb el sistema i sovint també independents de l'aplicació (transportables a altres aplicacions).



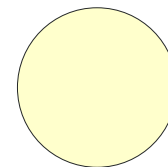
FormulariMatricula

Comunicació



GestorMatriculacio

Control



Assignatura





Entitat

$$[\text{visibilitat}] \text{ nom } [: \text{tipus}] [= \text{valor\_inicial}] [\{\text{propietats}\}]$$


{derived} {readOnly}

static









+ = pública  
 # = protegida  
 - = privada

Client	
	Nom : String
	DataAlta : Date = 01-gen-2000
	<u>DarrerCodiAssignat : Integer</u>
	CodiClient : Integer = DarrerCodiAssignat

[visibilitat] nom [(llista\_params)] [: tipus\_retorn] [{propietats}]


 + = pública  
 # = protegida  
 - = privada

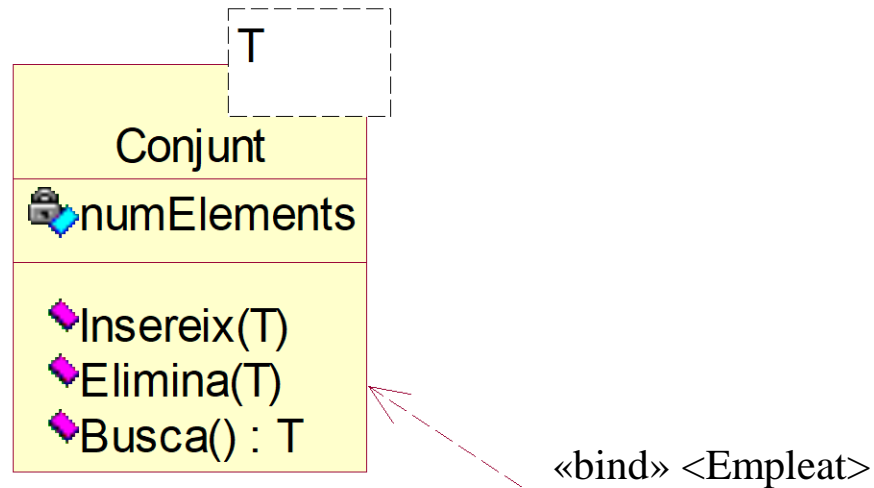
{isQuery}, {sequential}, {concurrent}

CompteCorrent	
	Titular
	Saldo
	Codi
	Reintegrament(valor : Currency)
	Ingres(valor : Currency = 10€) : Boolean
	UltimesOperacions()
	Saldo() : Currency
	Transferencia(valor : Currency, compteDesti : CompteCorrent) : Boolean

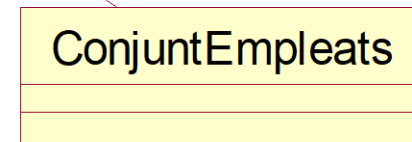
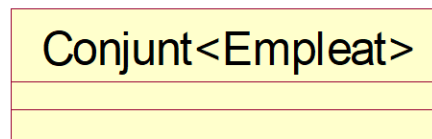


# Classes parametritzades

Classe Parametritzada



Instanciacions

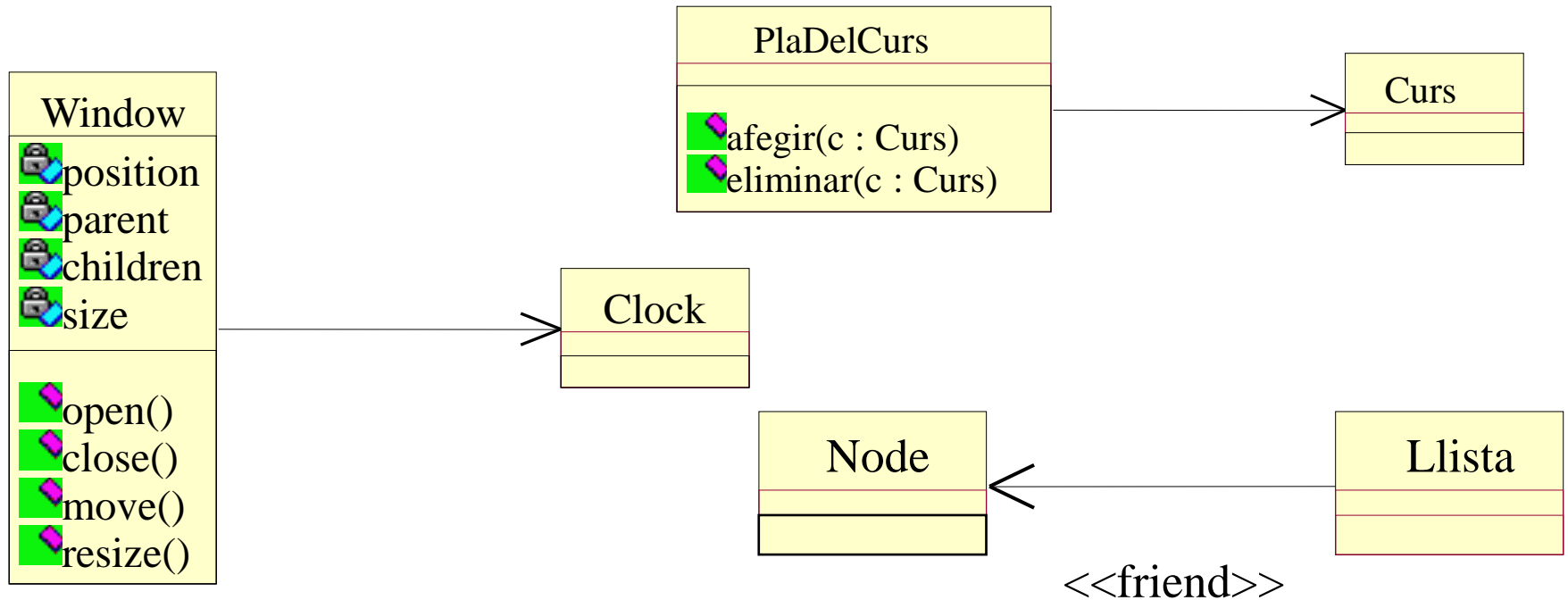


## Diagrames de classes: relacions

- Dependència
- Generalització
- Associació
- Agregació
- Composició



Un canvi en l'especificació d'un element afecta l'altre

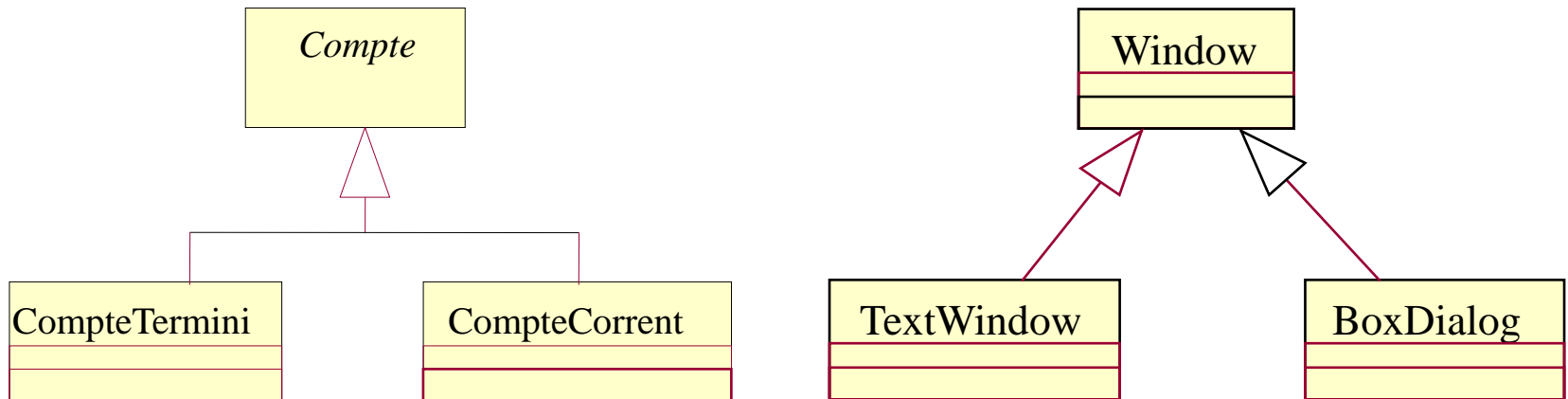


## Estereotipus per a dependències

---

- **bind:** entre una classe genèrica i una instanciació
- **friend:** dependència de classe amiga
- **refine:** relació de refinament
- **use:** relació d'ús
- **import:** un paquet importa els elements d'un altre.
- **extend:** per a casos d'ús
- **include:** per a casos d'ús

- Representa la relació “*ser un tipus de*”. És una relació entre classes on la classe especialitzada comparteix l'estructura i/o el comportament de la classe més general.
- Defineix una jerarquia d'abstraccions on la sub-classe (classe derivada) **hereta** les propietats d'una o més super-classes.

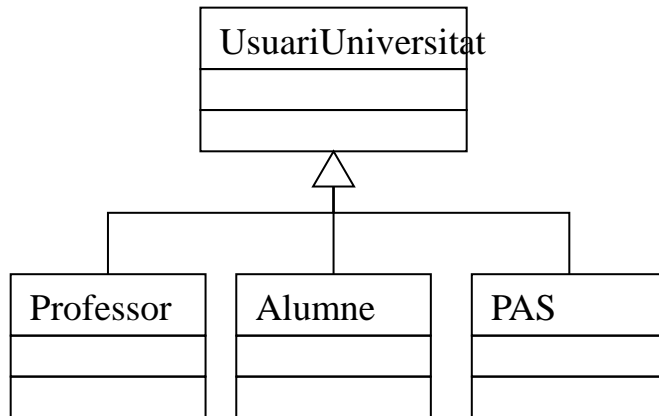


## Restriccions semàntiques entre les subclasses

---

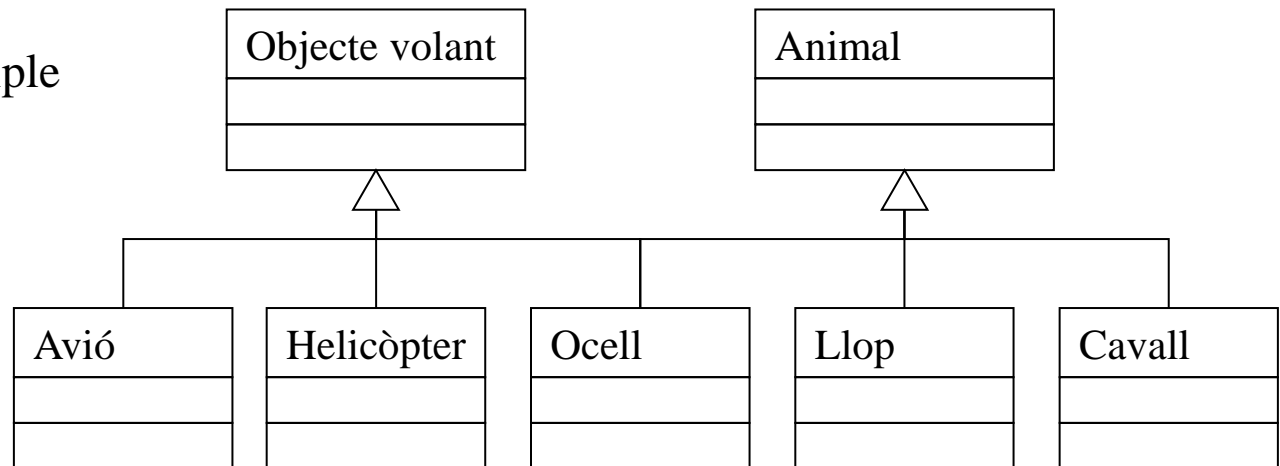
- Herència simple / herència múltiple:
  - **Herència simple:** una nova classe només deriva d'una classe.
  - **Herència múltiple:** una subclasse pot derivar de més d'una classe.
- Herència solapada / herència disjunta:
  - **Herència solapada:** una nova classe pot derivar de més d'una subclasse.
  - **Herència disjunta:** una nova classe no pot ser subclasse de més d'una subclasse.
- Herència completa / herència parcial:
  - **Herència completa:** es defineix una partició en què tots els objectes possibles estan representats per alguna subclasse.
  - **Herència parcial:** hi ha objectes no representats per les subclasses.
- Classificació dinàmica:
  - un objecte pot canviar de classe dins la jerarquia de subclasses.

## Herència simple / herència múltiple

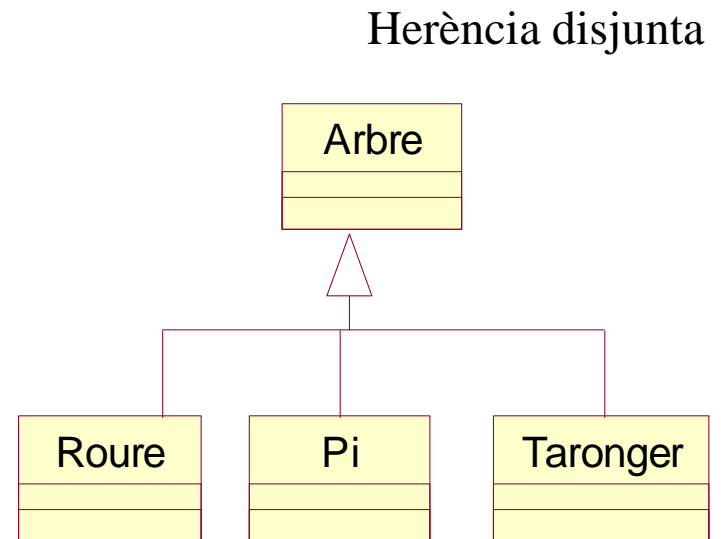
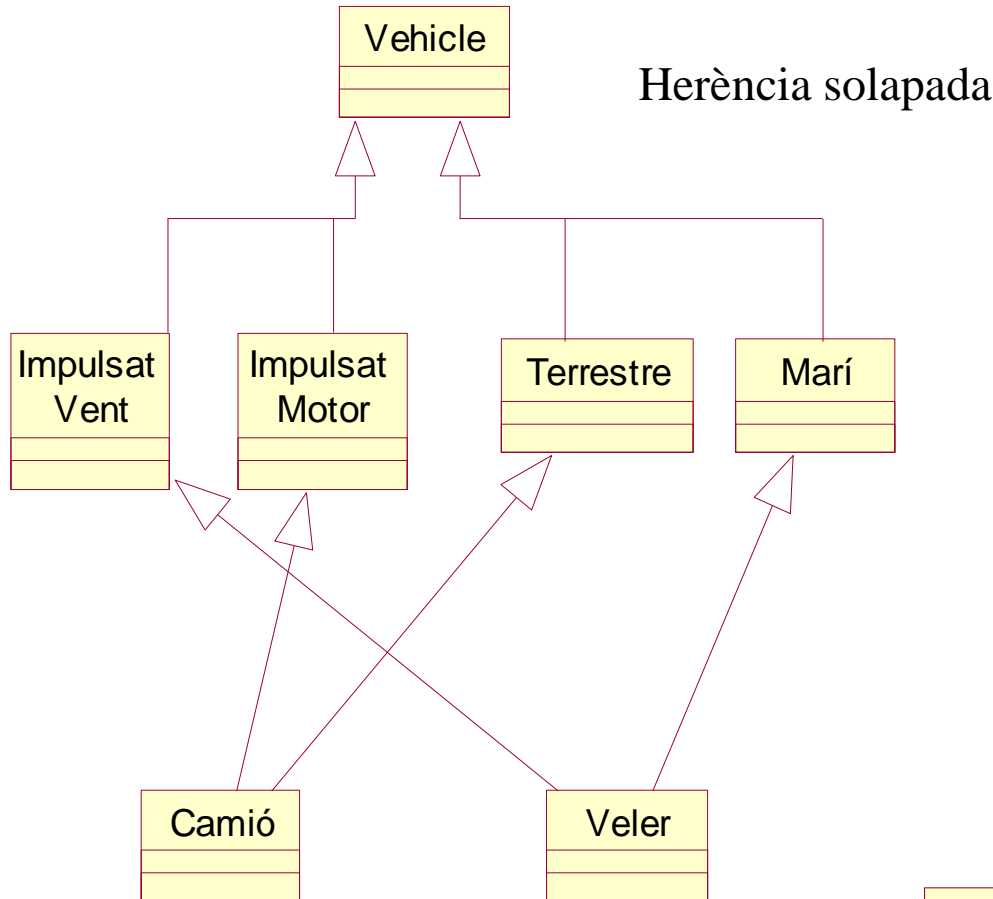


Herència simple

Herència múltiple

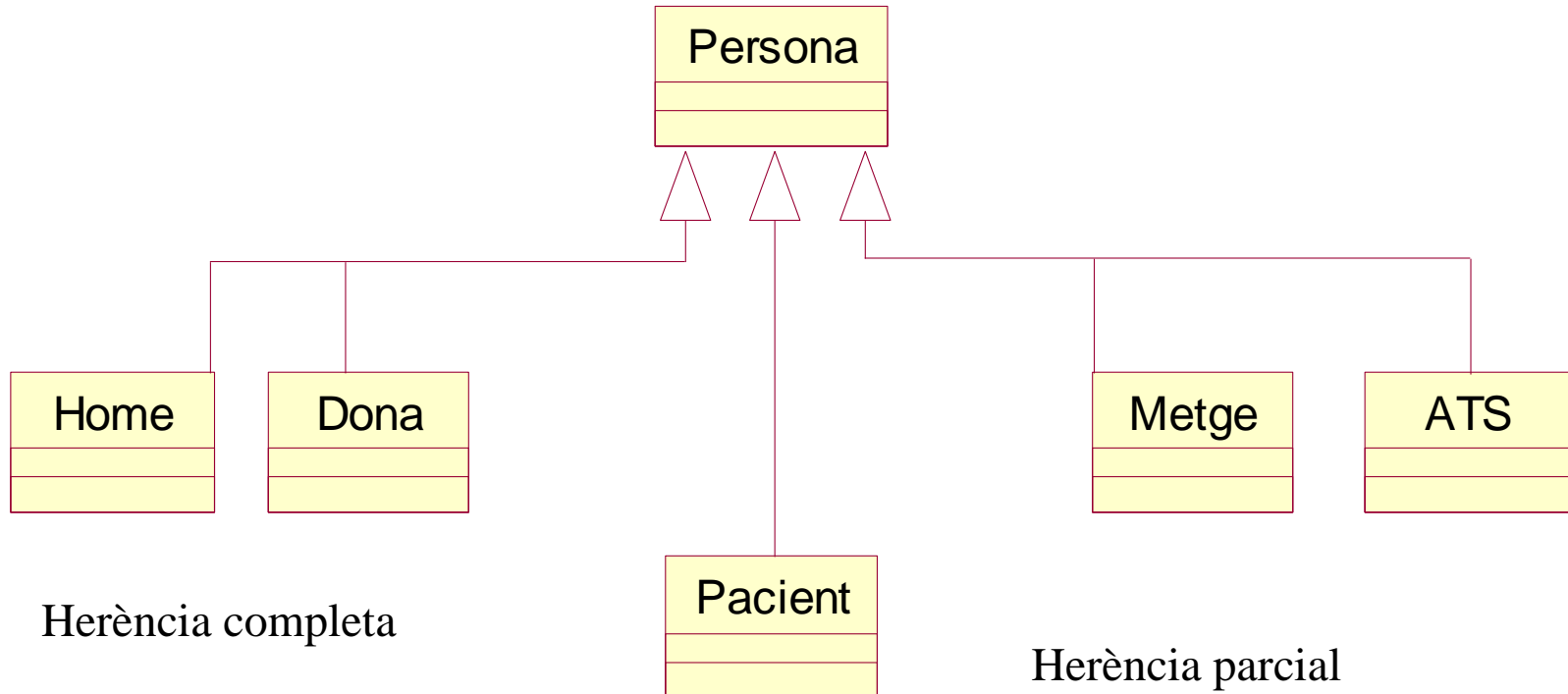


## Herència solapada / herència disjunta

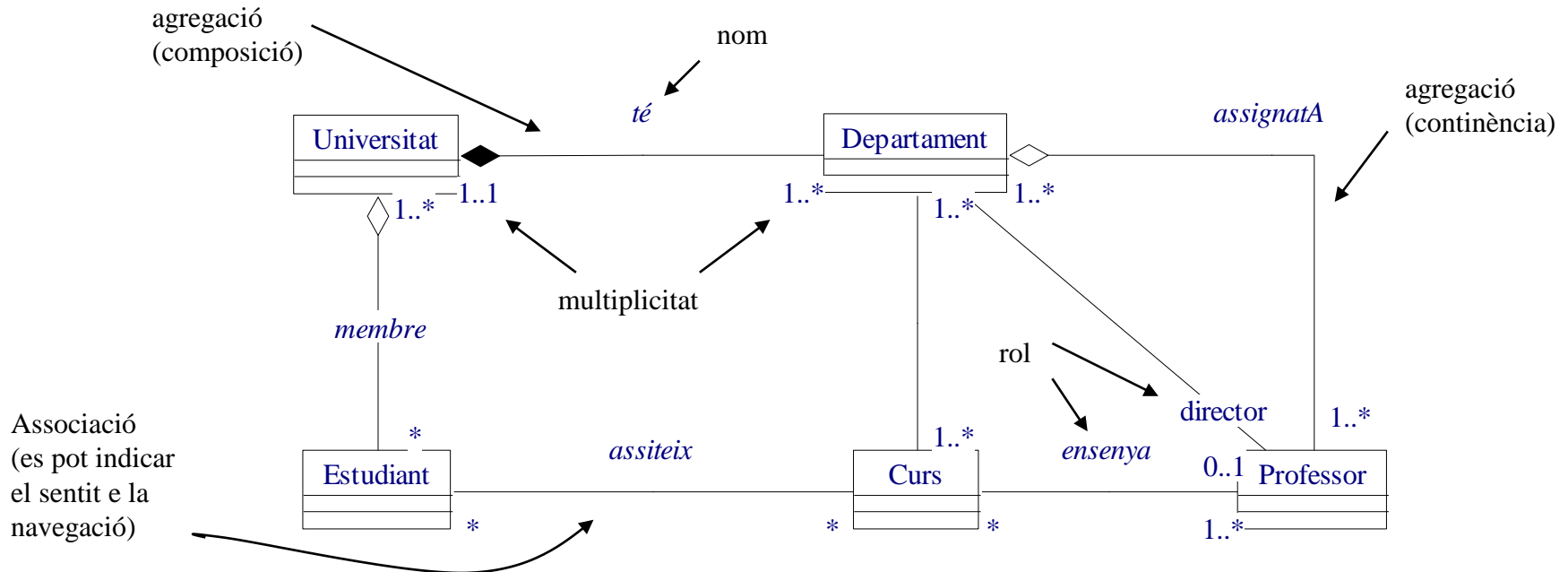




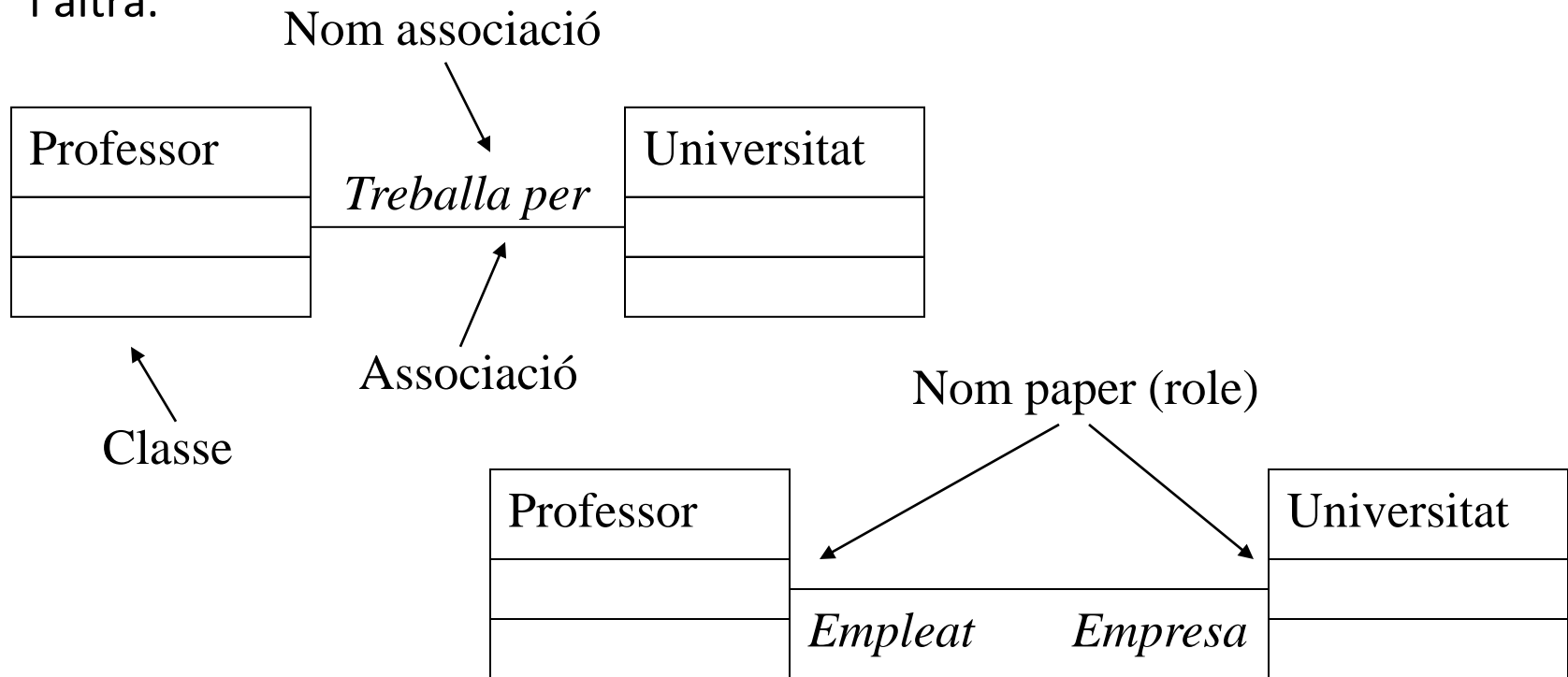
## Herència completa / herència parcial



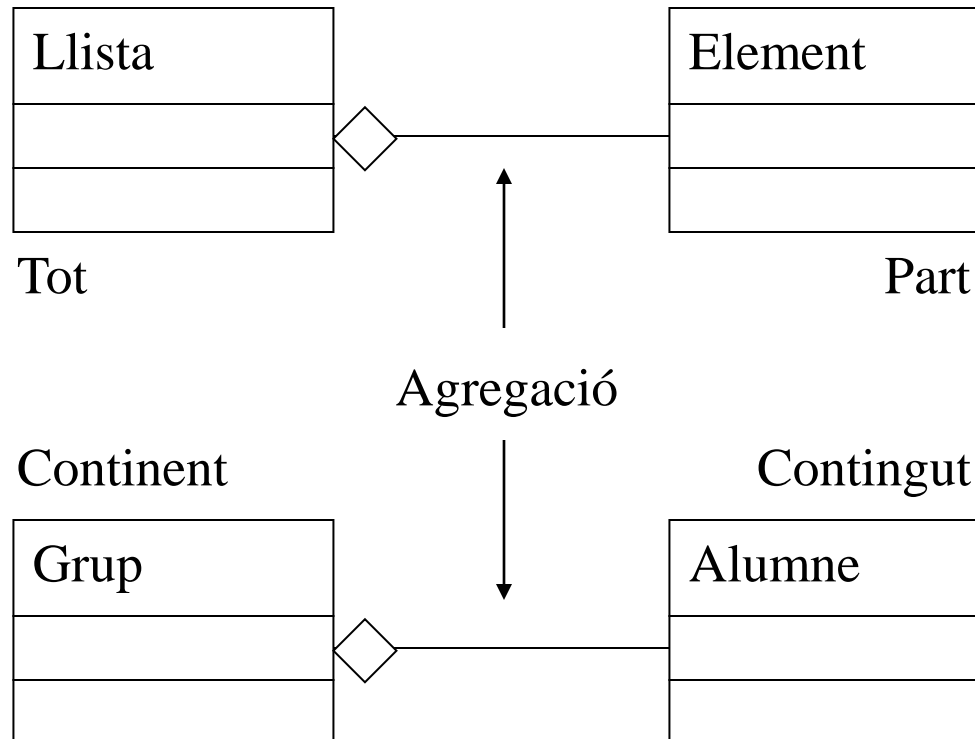
- Associació. Relació estructural que especifica que els objectes d'un element estan connectats als objectes d'un altre. Es pot navegar des d'un objecte d'una classe a un de l'altra.
- Agregació. És una associació *tot-part* o de *continent-contingut*.



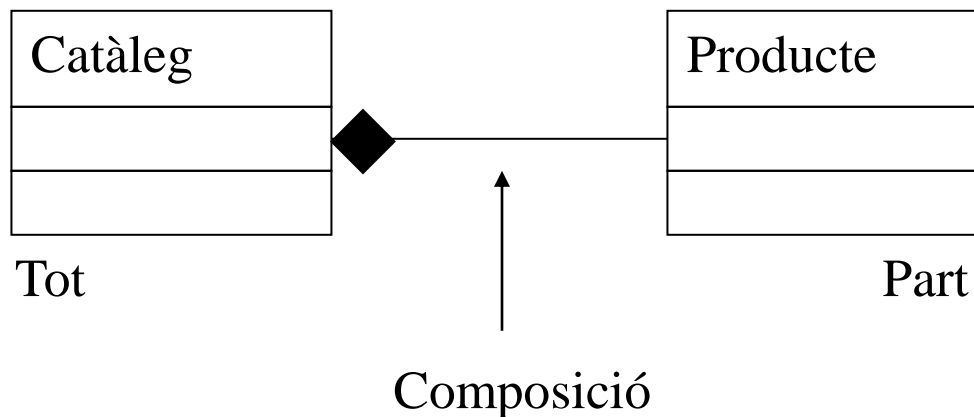
Modela una connexió semàntica entre classes. És una relació estructural que especifica que els objectes d'una classe estan connectats als objectes d'un altre. Es pot navegar des d'un objecte d'una classe a un de l'altra.



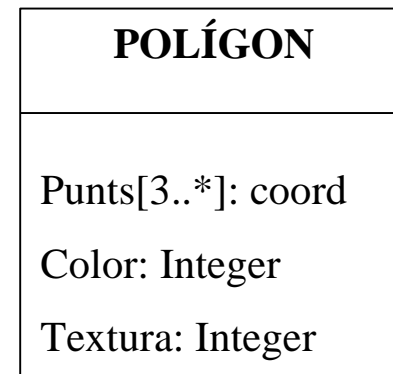
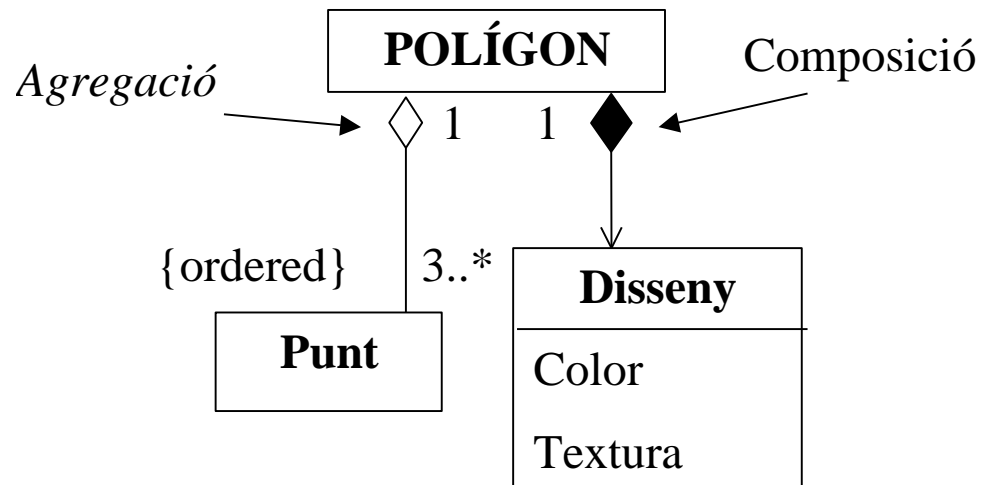
Una forma especial d'associació que modela una relació “tot-part” o de “continent-contingut” entre un agregat (el tot) i les seves parts.



- Una forma d'agregació amb un propietari fort i temps de vida coincidents. Les parts no poden sobreviure al tot (agregació).
- Les parts poden crear-se després de l'agregat a què pertanyen, però una vegada creades viuen i moren amb l'agregat.
- La part solament pot formar part d'un agregat.
- L'agregat gestiona la creació i destrucció de les parts.
- Les parts es poden eliminar abans d'eliminar l'agregat.



## Agregació i Composició: Exemple



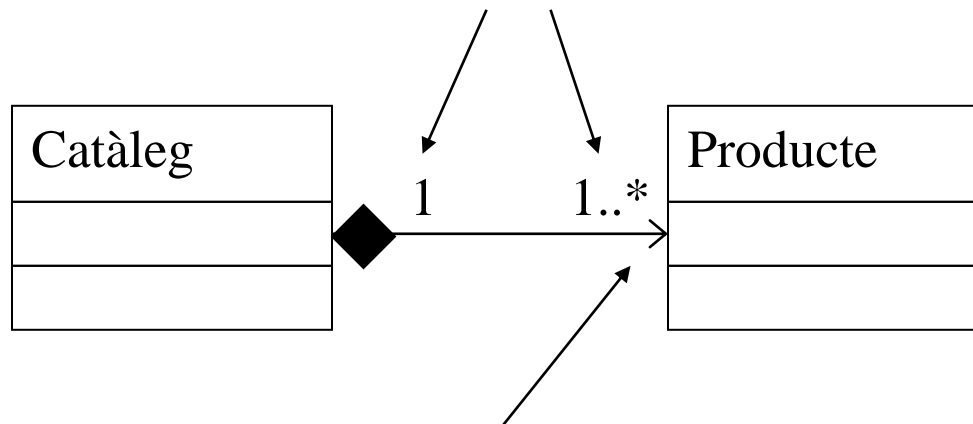
## Associació: Multiplicitat i Navegació

---

- **Multiplicitat.** La multiplicitat defineix quants objectes participen en una relació.
  - El número d'instàncies d'una classe relacionades amb UNA instància de l'altra classe.
  - S'especifiquen amb un rang en cada final de l'associació.
- **Navegació.** Les associacions i agregacions són bidireccionals per defecte, però de vegades és desitjable restringir la navegació a una direcció.
  - Si la navegació és restringida, s'afegeix una línia amb fletxa per indicar la direcció de la navegació, això vol dir, des de quin objecte es pot accedir a l'altre.

## Associació: Multiplicitat i Navegació. Exemple

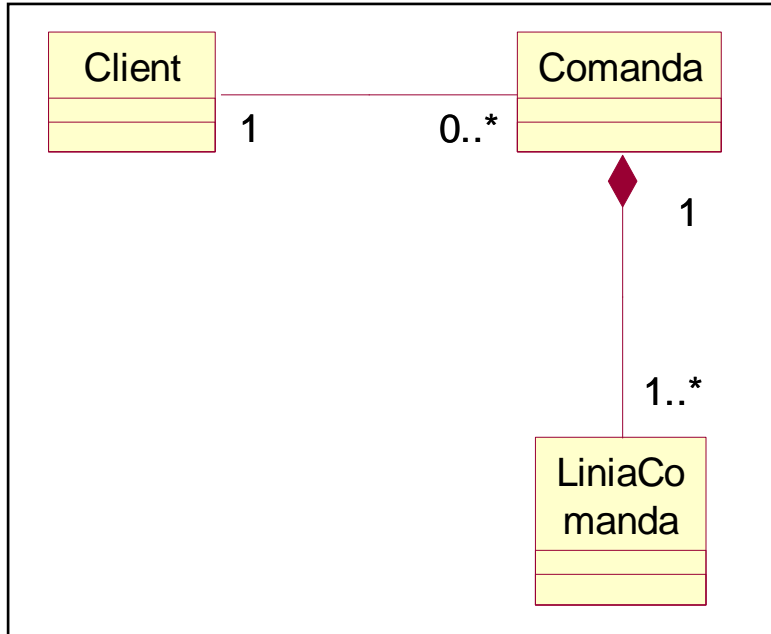
Multiplicitat (un catàleg conté entre 1 i n productes i un producte està contingut en un catàleg).



Navegació (des del catàleg podem accedir als productes però des dels productes no podem accedir al catàleg).



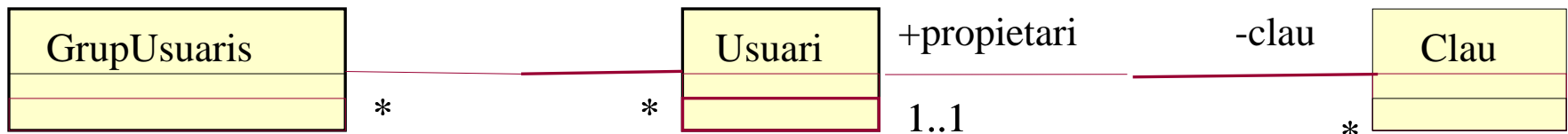
## Associacions: Implementació



```
class Comanda {
    Client _client;
    Set[liniaComanda] _liniesComanda;
    ...
public:
    getClient();
    getLiniesComanda();
}
```

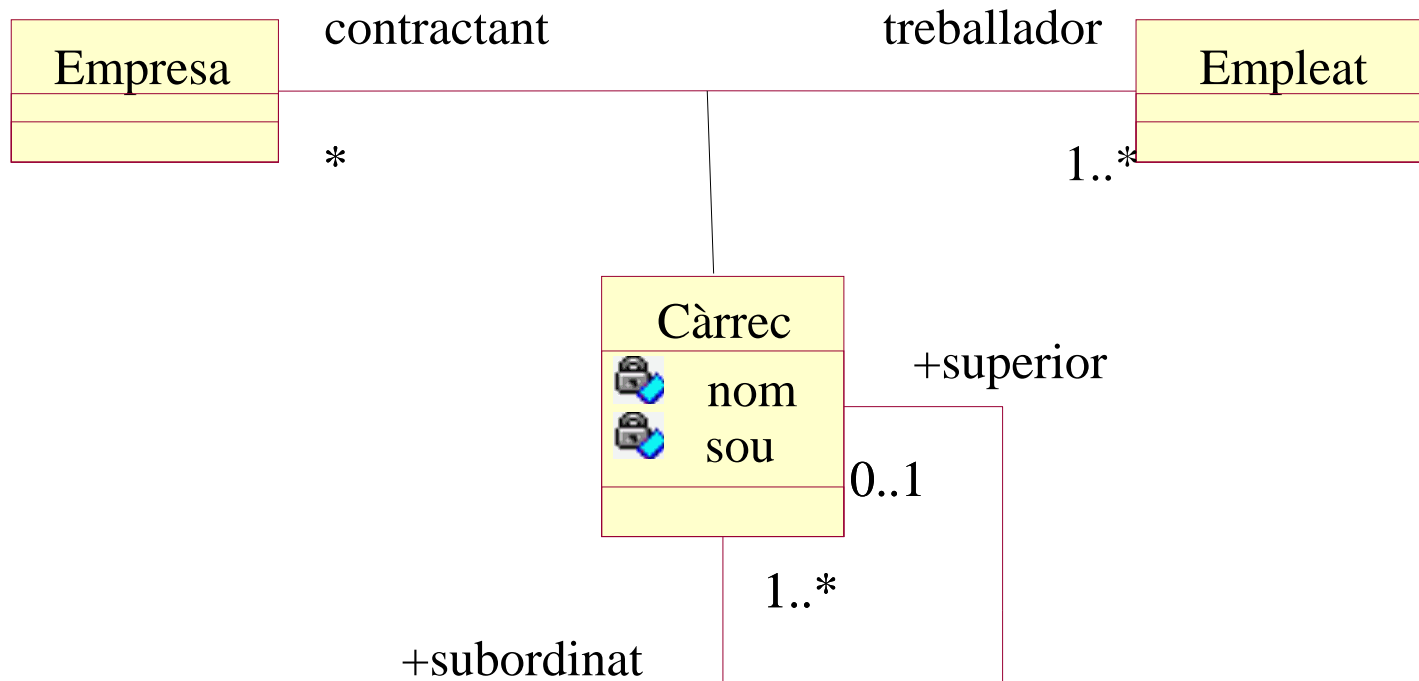
## Visibilitat en les associacions (rols)

- **public:** accessible per tots els clients.
- **protegit:** accessible només per classes derivades i si les classes de l'associació són “amigues”.
- **privat:** accessible només per la mateixa classe i si les classes de l'associació són “amigues”.

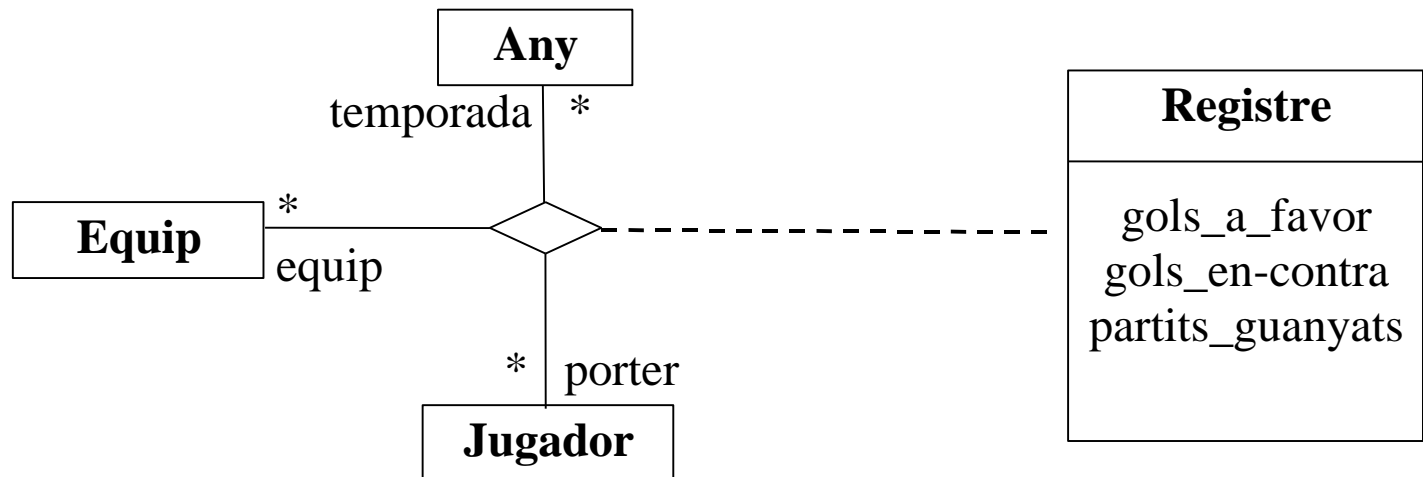


- Pública: +propietari
- Protegida: #propietari
- Privada: -propietari

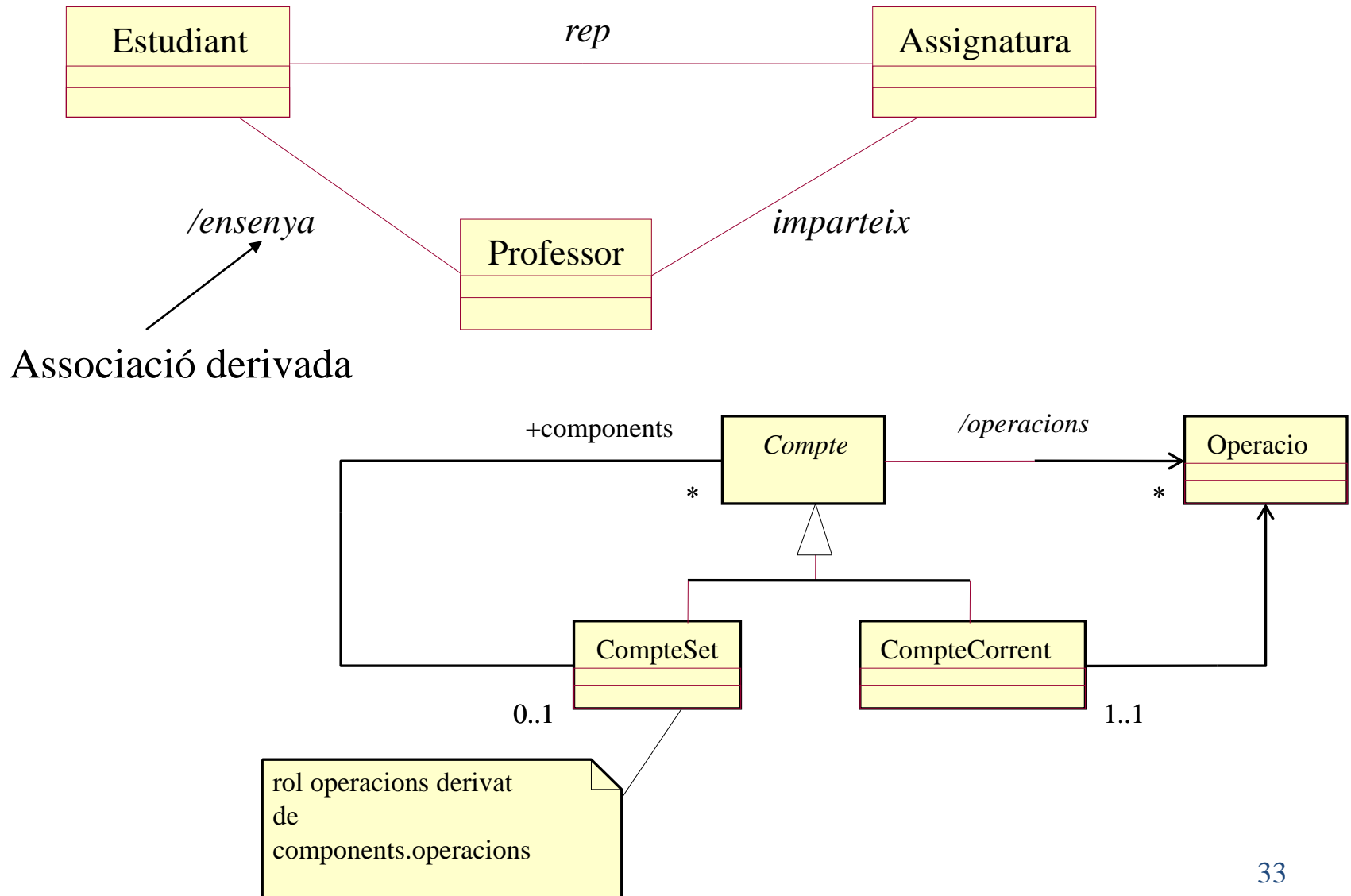
- Una classe associació afegeix una restricció: “Només pot existir una instància de l’associació entre qualsevol parell d’objectes participants”
- Una classe associació representa propietats que depenen dels dos objectes implicats en l’associació.



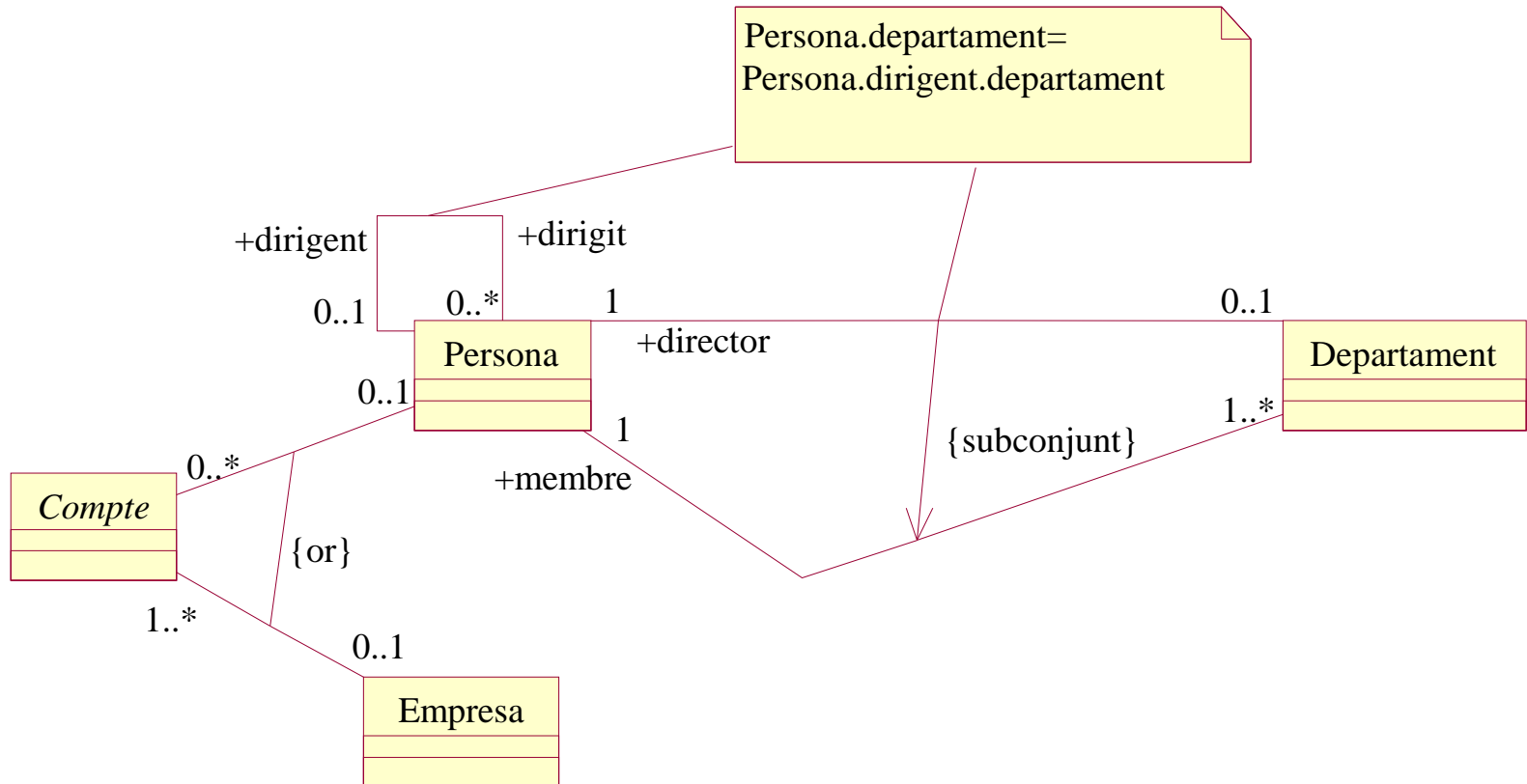
Associació entre tres o més classes: Cada instància de l'associació és una n-tupla de valors de cadascuna de les respectives classes.



## Associacions derivades

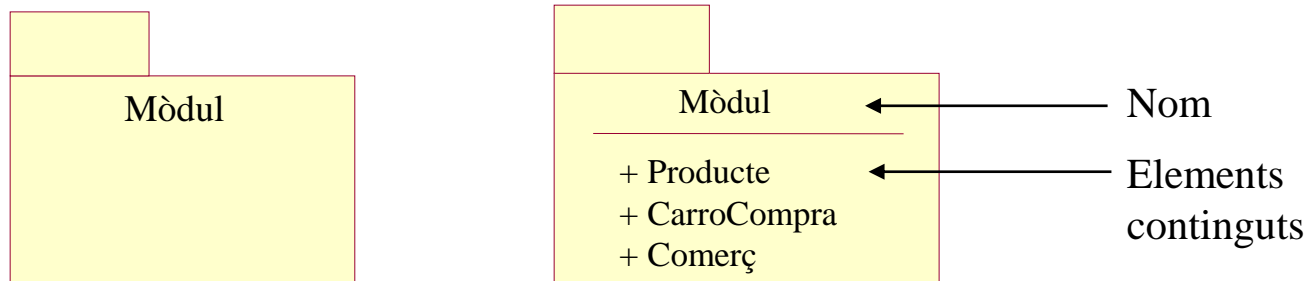


## Restriccions entre associacions



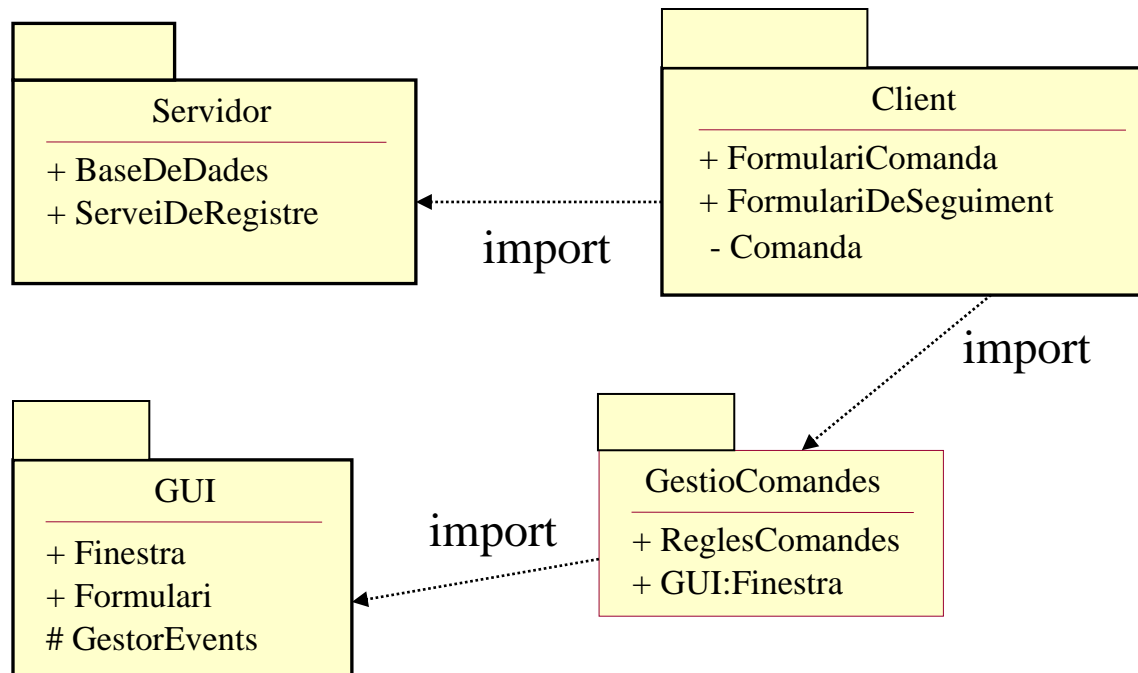
## Mòduls (paquets)

- És un element organitzatiu
- Poden contenir elements de qualsevol tipus, incloent altres paquets.
- Un element és exclusiu a un paquet.
- Els paquets han de mantenir la màxima cohesió (una única tasca) i el mínim acoblament (poca interdependència entre mòduls).
- Tot i poder contenir altres paquets, cal evitar un excés d'enniuament.



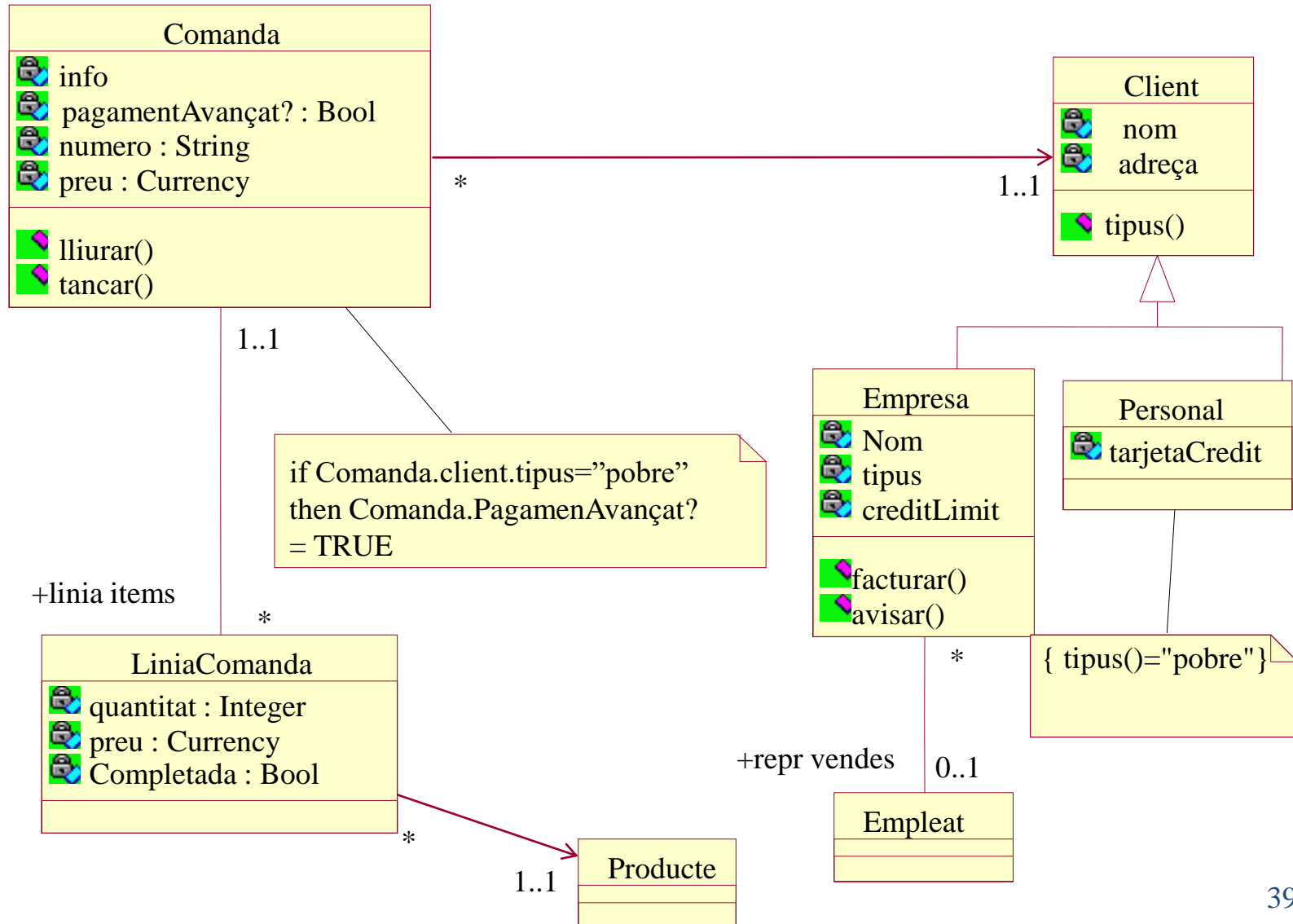
## Importació/Exportació de paquets

- Igualment que les classes, els paquets poden exportar una part pública que és importada pel paquet contenidor. Això permet gestionar diferents nivells d'abstracció.
- La importació no és transitiva.
- Els paquets continguts en altres paquets veuen tot allò que el paquet contenidor importa.

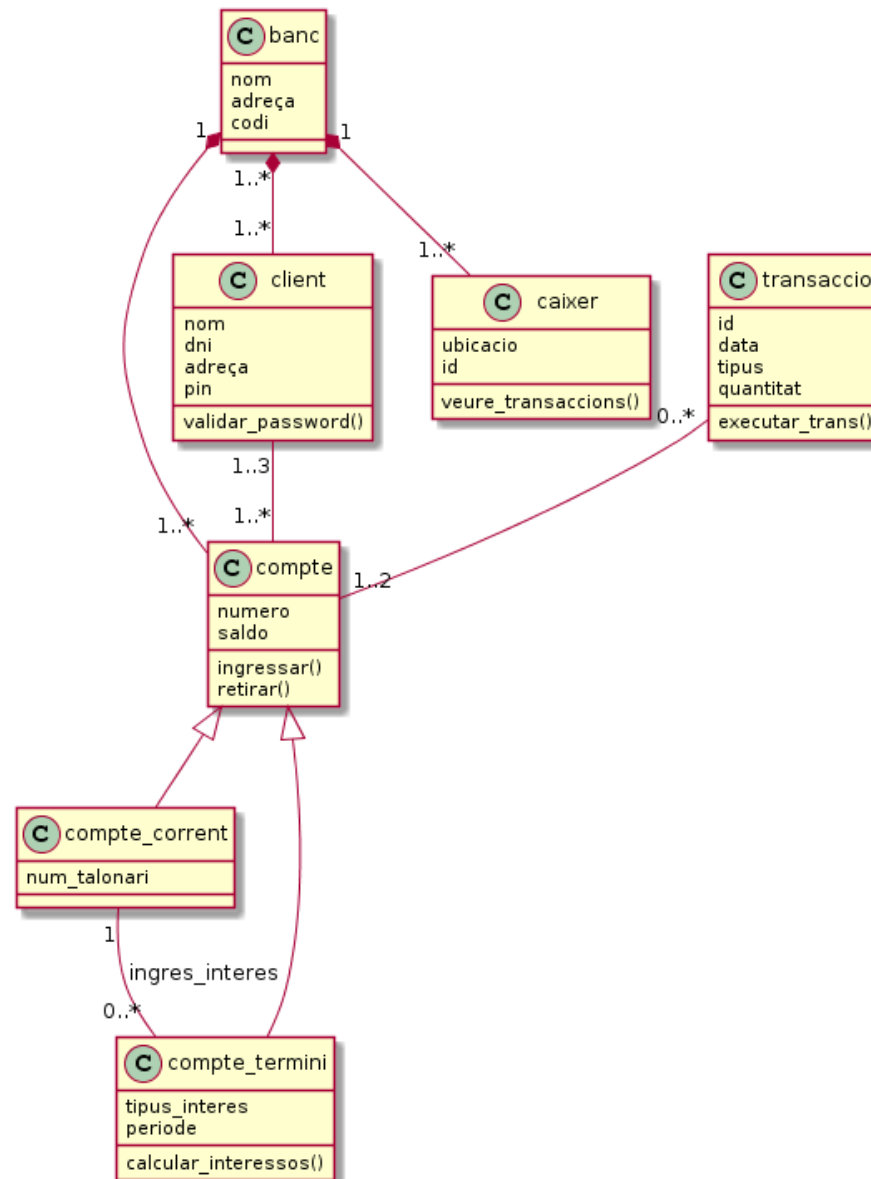




# Diagrama de classes: Exemple



## Diagrama de classes: Exemple



## Diagrama de classes: Exemple

