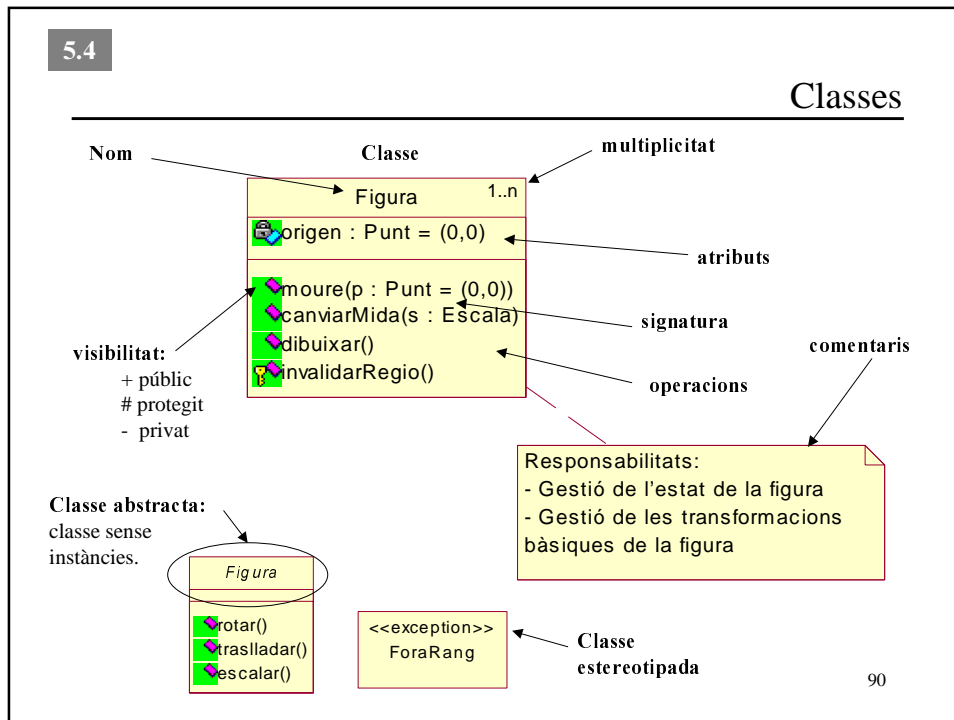


- Es descriuen els tipus d'objectes d'un sistema i les **relacions estàtiques** que hi ha entre ells.
- S'expressa mitjançant els **diagrames de classe**.
- Normalment contenen:
  - Classes
  - Interfícies
  - Relacions de dependència, realització, generalització i associació (agregació, composició)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

- **Classe:** descripció d'un grup d'objectes amb propietats comunes (atributs), comportament comú (operacions), relacions comunes amb altres objectes i semàntica comuna.
- Els diagrames de classes s'utilitzen per modelar:
  - **Vocabulari del sistema:** identificar classes per abstraccions rellevants del domini del problema.
  - **Col·laboracions (part estàtica):** identificar classes i interfícies la interacció dels quals produeix el comportament desitjat. Un diagrama de col·laboració representa una instància del diagrama de classes per a un escenari concret.
  - **Esquema lògic de bases de dades.**



### 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.

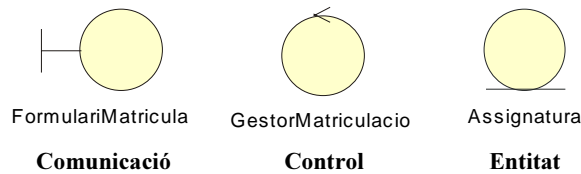
Notació →



## Classes d'anàlisi

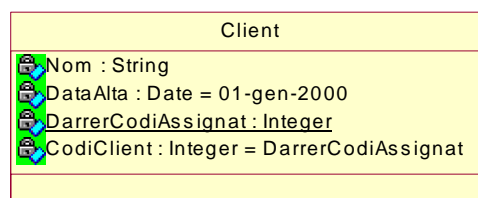
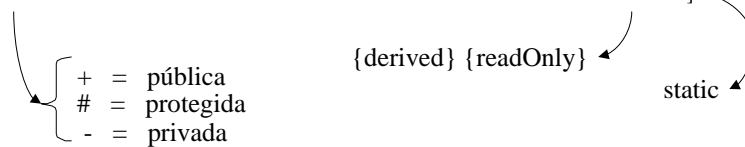
Les classes del model d'anàlisi són classes estereotipades que representen un model conceptual inicial per als elements del sistema que tenen responsabilitats i comportament. Hi ha tres tipus de classes d'anàlisi:

- **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).



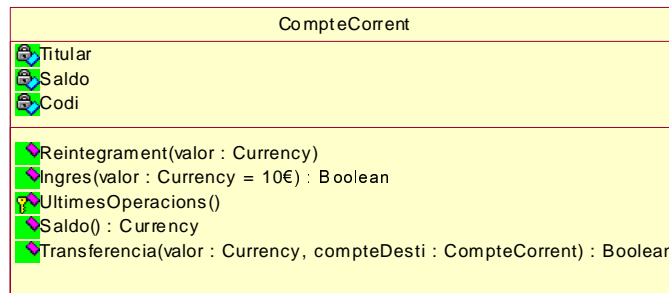
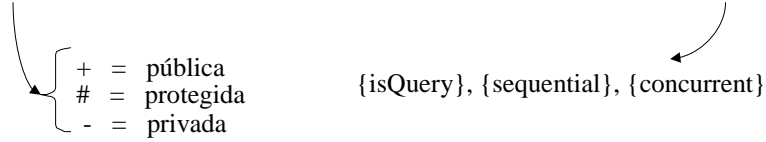
## Atributs

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



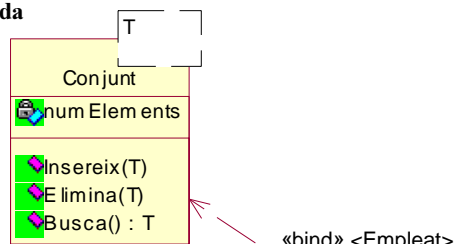
## Operacions

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

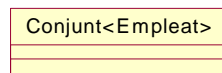


## Classes parametritzades

Classe Parametritzada



Instanciacions



5.4

## Diagrames de classes: relacions

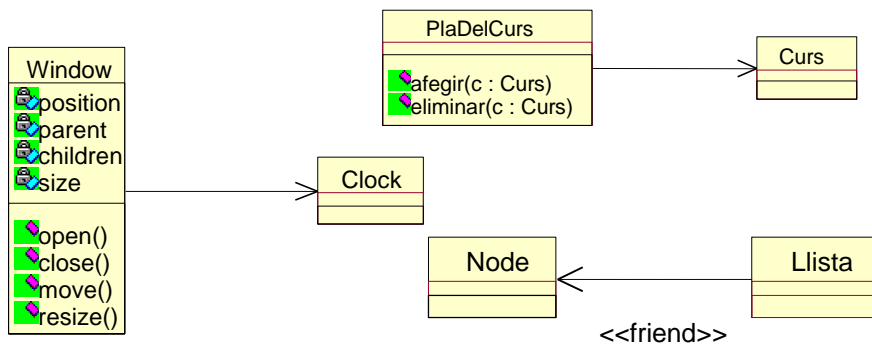
- Dependència
- Generalització
- Associació
- Agregació
- Composició

96

5.4

## Dependència

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



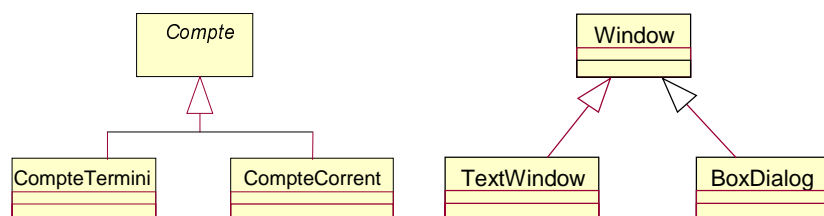
97

## Estereotipus per a dependències

- **bind:** entre una classe genèrica i una instanciació
- **friend:** dependència de classe amiga
- **refine:** relació de refinamiento
- **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

## Generalització

- 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.

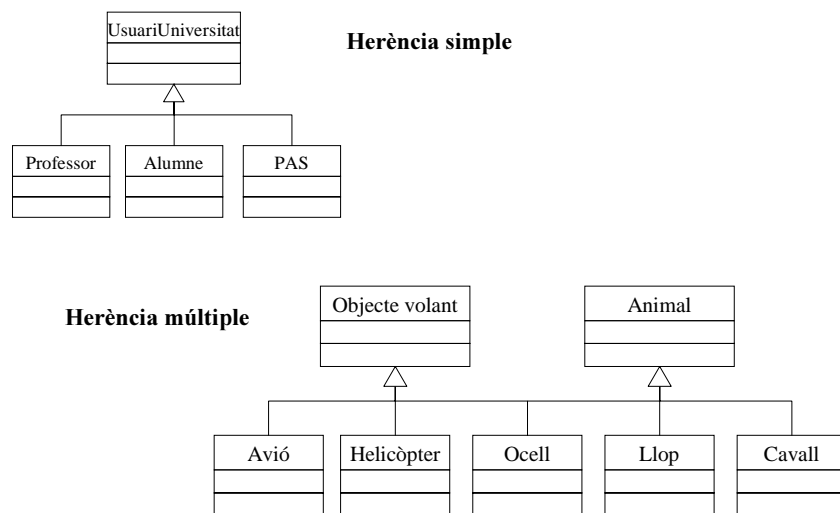


## Restriccions semàntiques entre les subclasses

- 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:
 

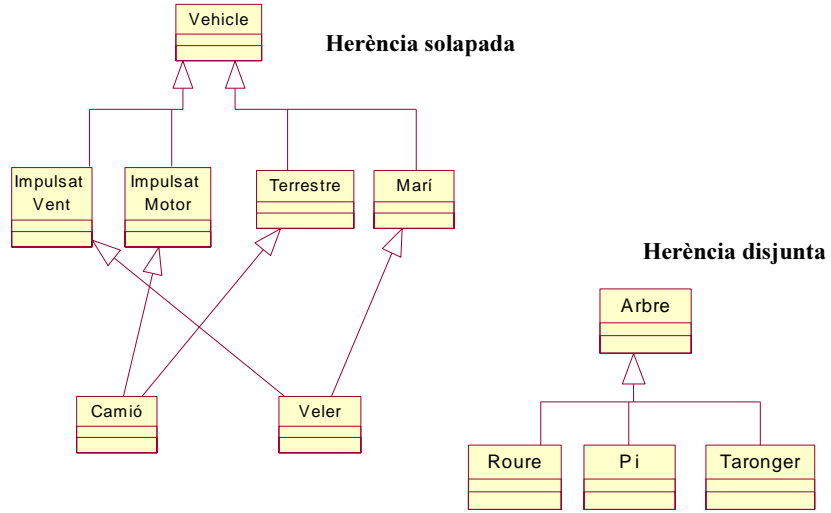
un objecte pot canviar de classe dins la jerarquia de subclasses.

## Herència simple / herència múltiple



5.4

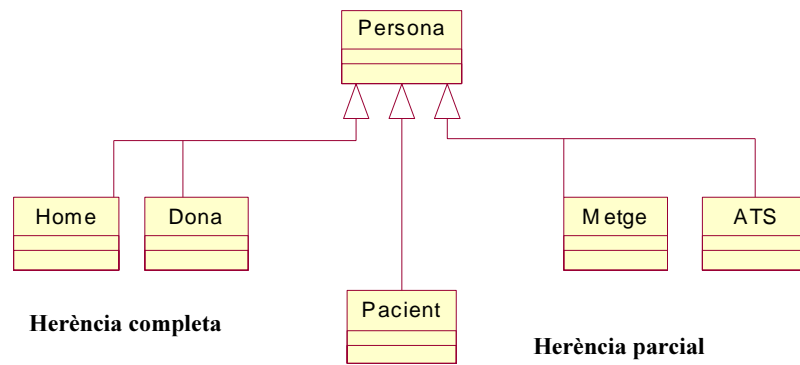
### Herència solapada / herència disjunta



102

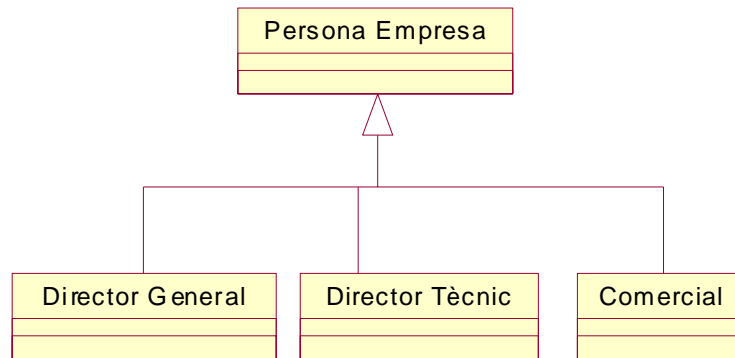
5.4

### Herència completa / herència parcial



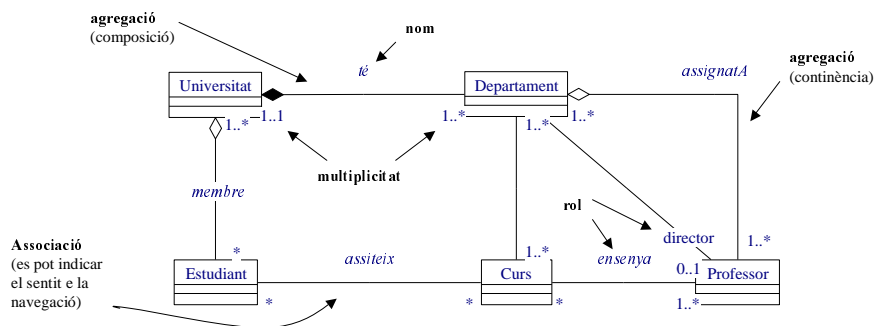
103

## Classificació dinàmica



## Associació

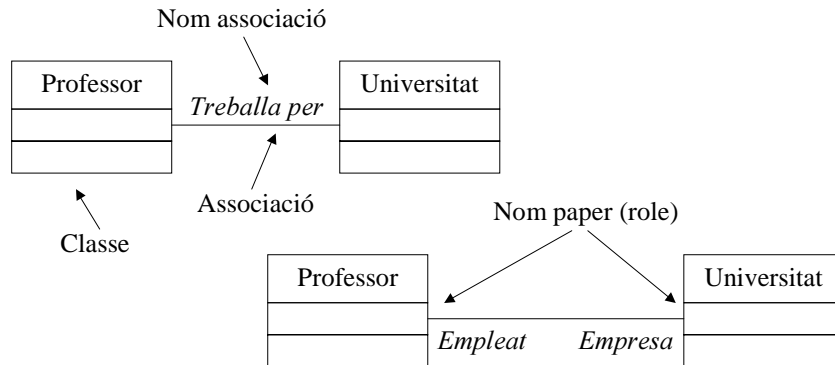
- **Associació.** Relació estructural que especifica que els objectes d'un element estan connectats als objectes d'un altre. Es pot navegar des d'un objecte d'una classe a un de l'altra.
- **Agregació.** És una associació *tot-part* o de *continent-contingut*.



5.4

## Associació

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.

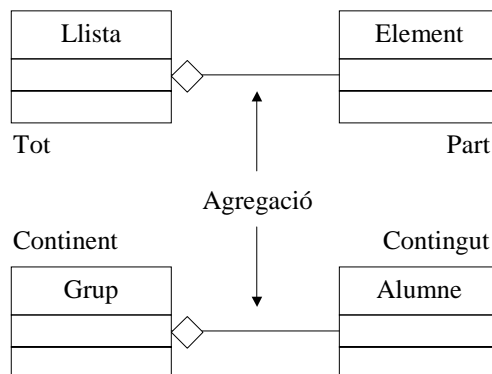


106

5.4

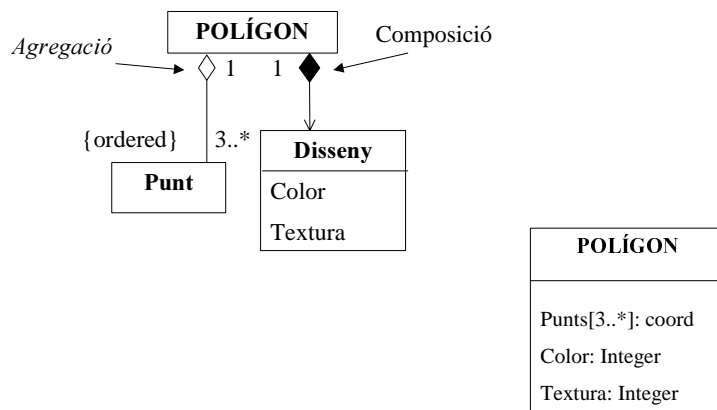
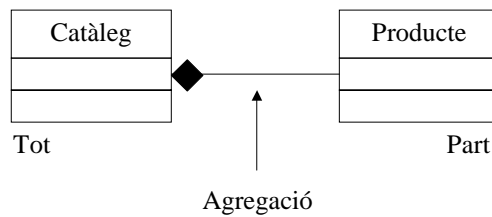
## Agregació

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



107

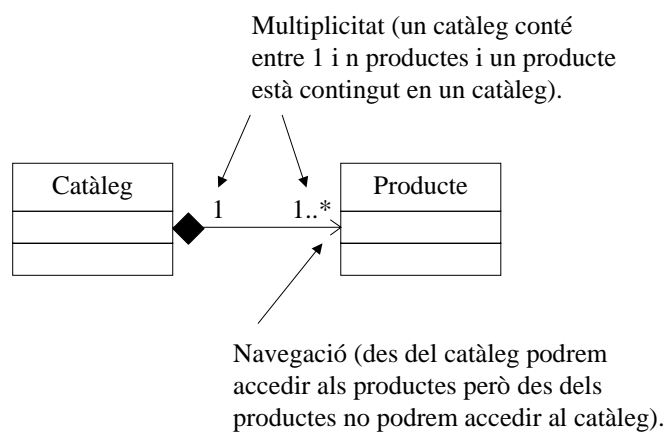
- Una forma d'agregació amb un propietari fort i temps de vida coincidents. Les parts no poden sobreviure al tot (agregació).
- Les parts poden crear-se després de l'agregat a què pertanyen, però una vegada creades viuen i moren amb l'agregat.
- La part solament pot formar part d'un agregat.
- L'agregat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



## Associació: Multiplicitat i Navegació

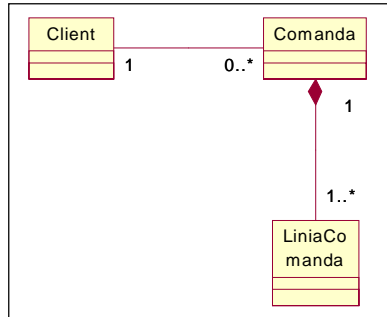
- **Multiplicitat.** La multiplicitat defineix quants objectes participen en una relació.
  - El número d'instàncies d'una classe relacionades amb UNA instància de l'altra classe.
  - S'especifiquen amb un rang en cada final de l'associació.
- **Navegació.** Les associacions i agregacions són bidireccionals 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



5.4

## Associacions: Implementació



```

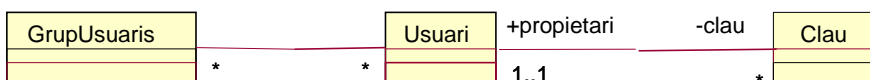
class Comanda {
  Client_client;
  Set[liniaComanda] _liniesComanda;
  ...
  public:
  getClient();
  getLiniesComanda();
}
  
```

112

5.4

## 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".

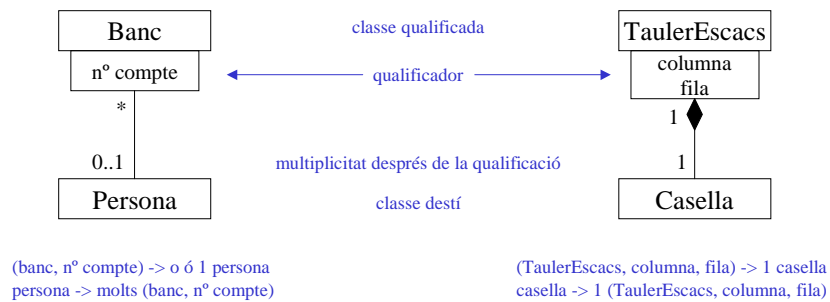


- Pública: +propietari
- Protegida: #propietari
- Privada: -propietari

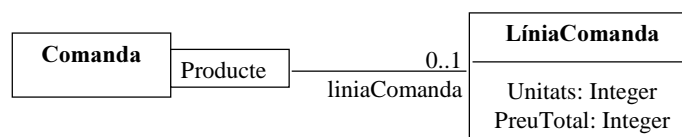
113

## Associació qualificada

- Denota un atribut o llista d'atributs en els quals el valor dels atributs seleccionen un únic objecte relacionat o un conjunt d'objectes relacionats dins de tot el conjunt d'objectes relacionats segons aquesta relació.
- Es tracta d'un índex per recórrer aquesta associació (es podria entendre com si els elements es guardessin en una taula).



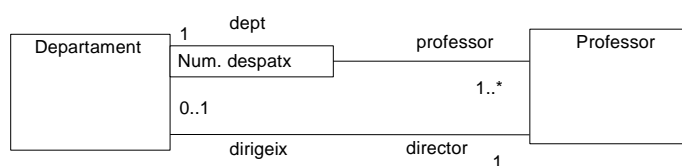
## Associacions qualificades: Exemples



```

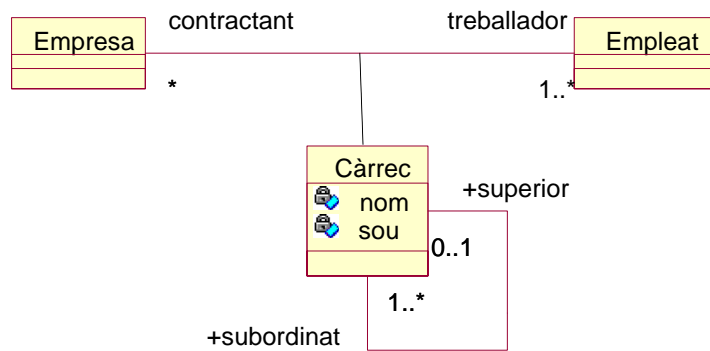
Class Comanda {
    private Table _liniesComanda;
    public LiniaComanda getLiniaComanda(Producte unProducte);
    public void addLiniaComanda (Integer quantitat, Producte elProducte);
    ...
}

```



## Classes Associació

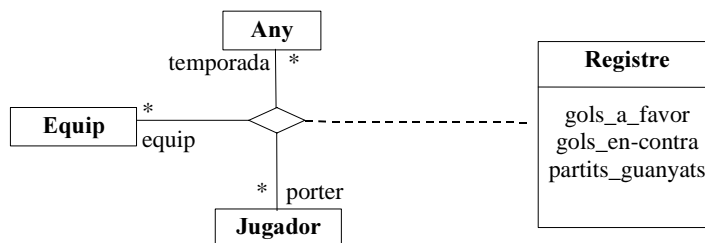
- Una classe associació afegeix una restricció: “Només pot existir una instància de l’associació entre qualsevol parell d’objectes participants”
- Una classe associació representa propietats que depenen dels dos objectes implicats en l’associació.



116

## Associacions n-aries

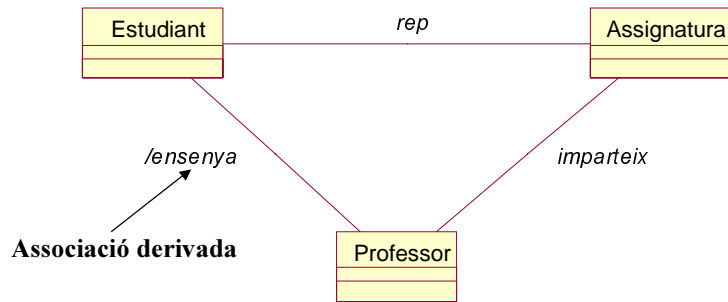
Associació entre tres o més classes: Cada instància de l’associació és una n-tupla de valors de cadascuna de les respectives classes.



117

5.4

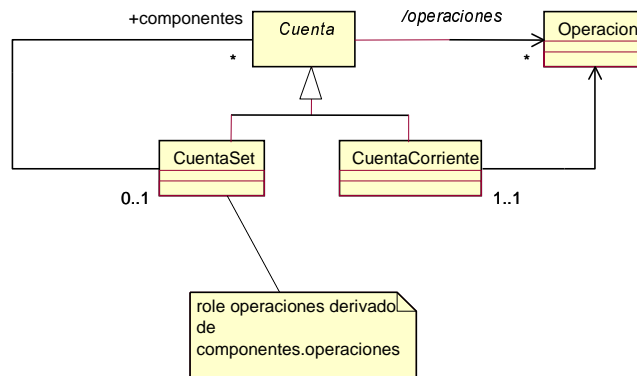
## Associacions derivades



118

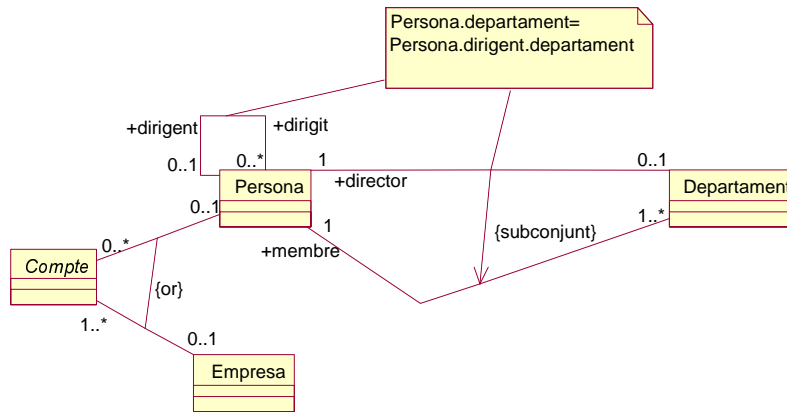
5.4

## Asociaciones derivadas



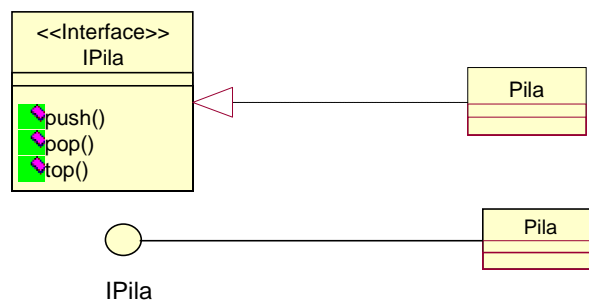
119

## Restriccions entre associacions



## Diagrames de Classe: Realització

Una classe que és realització d'una altra indica que la primera garanteix la implementació de l'especificació de la segona.

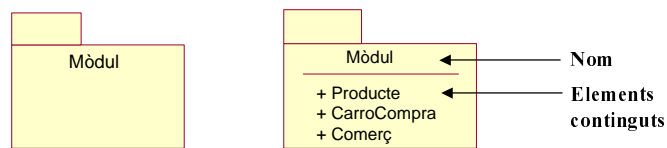


**Interface:** representa la visió externa de les funcionalitats de l'objecte que s'exporten. És una col·lecció d'operacions que s'utilitza per especificar un servei d'una classe o una component.

**Realització:** representa la implementació d'allò especifica la interfície.

## Mòduls (paquets)

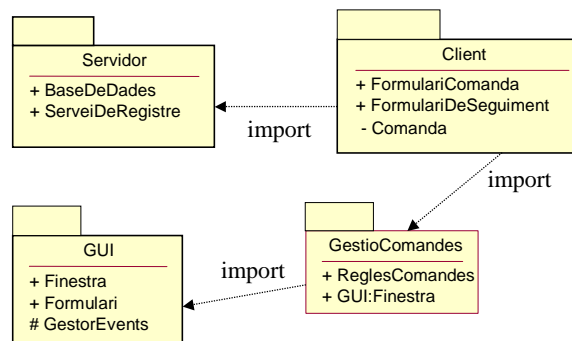
- És un element organitzatiu
- Poden contenir elements de qualsevol tipus, incloent altres paquets.
- Un element és exclusiu a un paquet.
- Els paquets han de mantenir la màxima cohesió (una única tasca) i el mínim acoblament (poca interdependència entre mòduls).
- Tot i poder contenir altres paquets,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



122

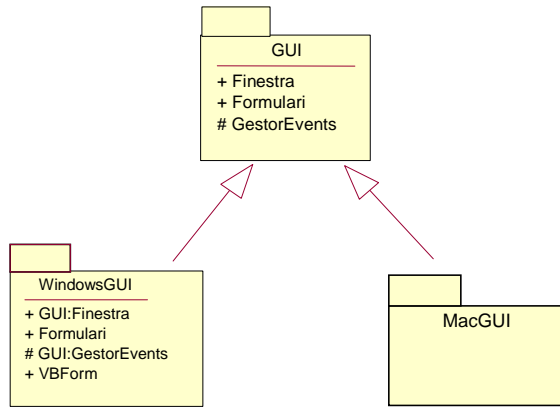
## Importació/Exportació de paquets

- 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.

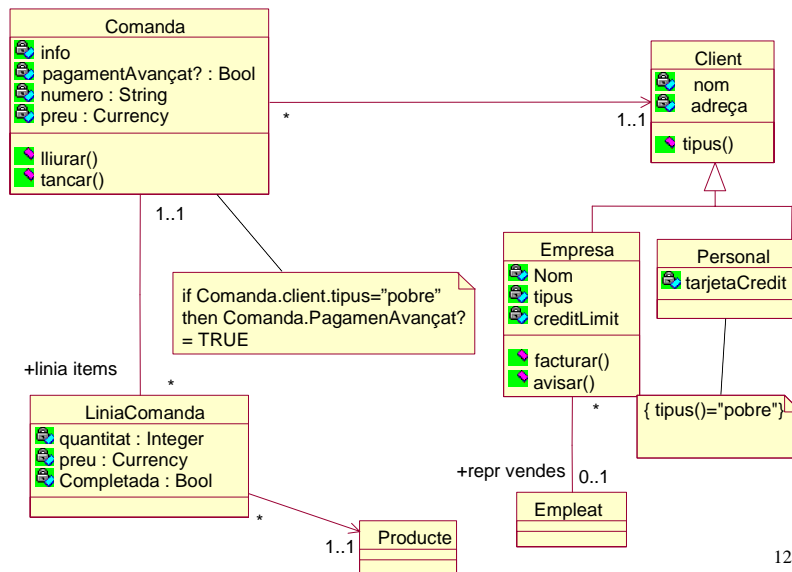


123

### Generalització de Paquets

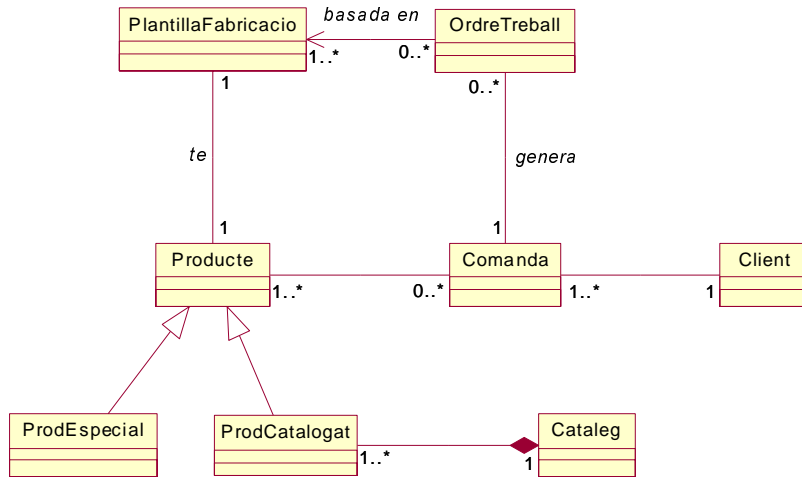


### Diagrama de classes: Exemple



5.4

## Diagrama de classes: Exemple

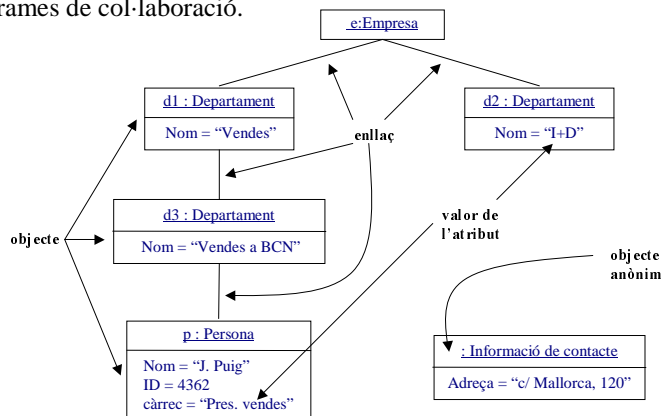


126

5.4

## Diagrama d'objectes

- Representa un conjunt d'objectes i relacions en un moment concret.
- És una “fotografia” del sistema en un moment d'execució. Instància del diagrama de classes.
- Representa una visió estàtica del sistema. Complementat amb diagrames de col·laboració.

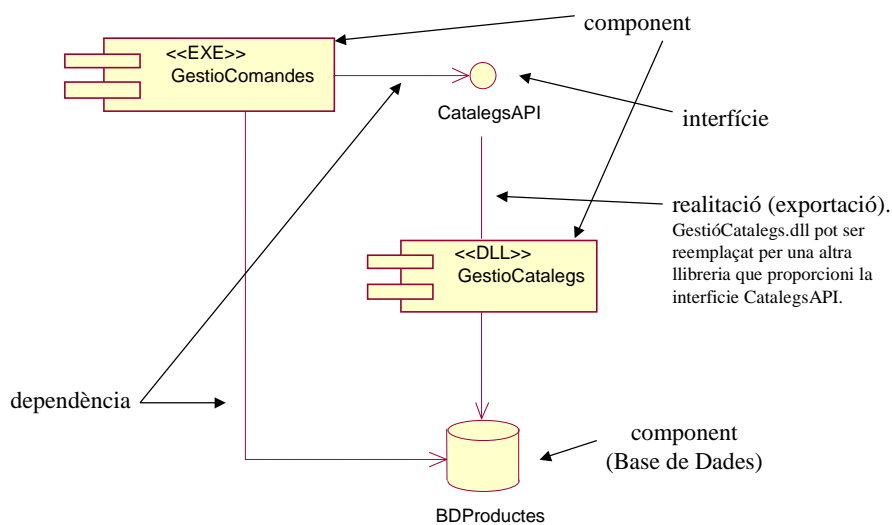


127

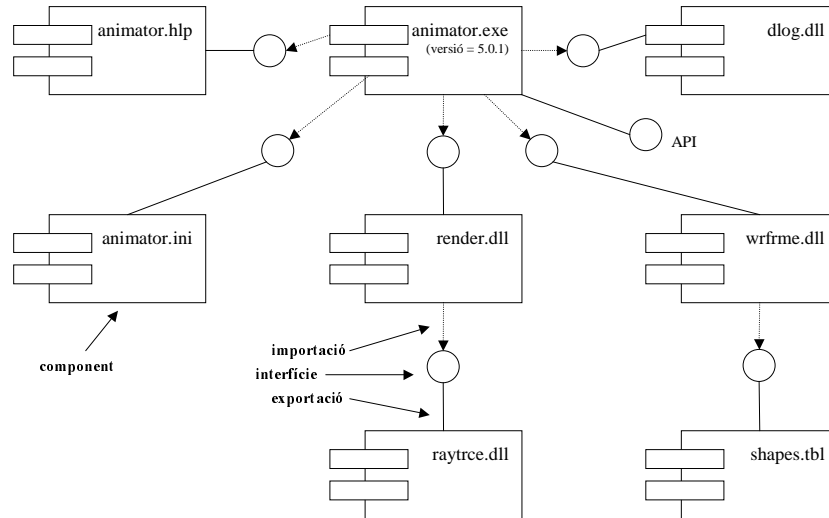
## Diagrama de components

- Una component és una part física i reemplaçable d'un sistema que realitza un conjunt d'interfícies i proporciona la realització d'aquestes.
- Modela artefactes com ara: executables, biblioteques, taules, fitxers, documents, etc.
- Representa l'empaquetament físic d'elements lògics com ara classes, interfícies, etc.
- Tipus de components:
  - Desplegament. Components necessàries per formar el sistema executable (DLL, EXE, etc..).
  - Producte de treball. Productes que queden al final del procés de desenvolupament (codi font, fitxers de dades, etc..).
  - Execució. Components que es creen com a conseqüència d'un sistema en execució (objecte COM instanciat d'una dll).

## Diagrama de components. Notació



## Diagrama de components. Exemple



130

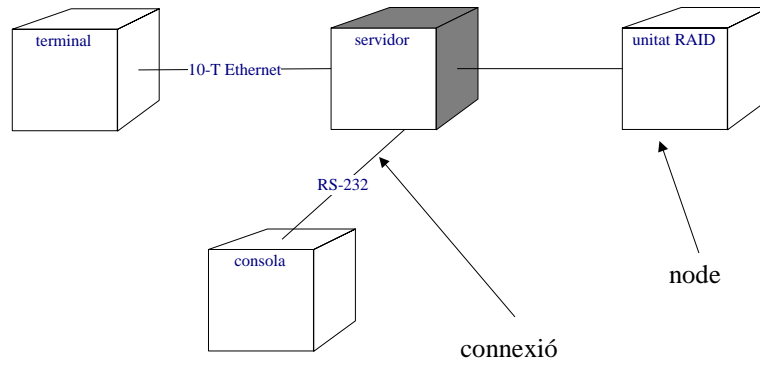
## Diagrama de desplegament

- Dimensió lògica d'un sistema: classes, interfícies, col·laboracions, interaccions i màquines d'estats.
- Dimensió física d'un sistema:
  - Components: empaquetaments físics dels elements lògics.
  - Nodes:
    - Elements físics que existeixen en temps d'execució i representen un recurs computacional que pot tenir memòria i capacitat de processament.
    - Hardware sobre el qual es despleguen i executen les components.
  - Arcs: connexions físiques entre nodes.
- Els diagrames de desplegament mostren la configuració dels nodes que participen en l'execució i de les components que resideixen als nodes.

131

5.5

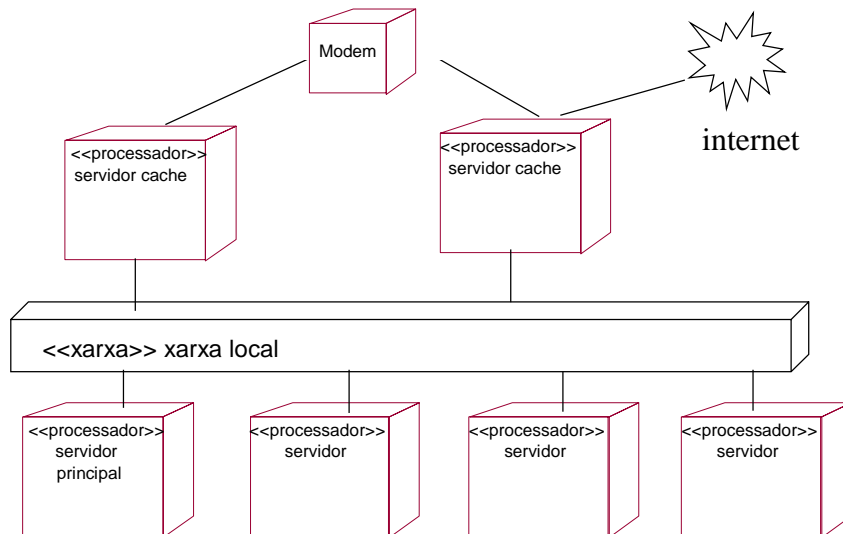
### Diagrama de desplegament. Notació



132

5.5

### Diagrama de desplegament. Exemple



133