INSTITUT SABADELL 2020-2021

DISSENY DE PROGRAMARI AMB UML

1r DAM / DAM-vi

Índex

- Origen de l'UML
- Utilitat de l'UML
- Casos d'ús.
- Diagrames de classe.
- Diagrames de seqüència.

DISSENY DE PROGRAMARI AMB UML

Origen de l'UML.

- L'anàlisi i disseny del programari constitueix una part fonamental en qualsevol projecte, especialment entre les parts de presa de requisits i desenvolupament de projecte.
- UML va néixer a mitjans dels anys 90 a l'empresa Rational Software, com a una iniciativa per a desenvolupar un estàndard de modelatge.
- Posteriorment es van afegir a aquesta iniciativa científics i altres empreses del sector.

DISSENY DE PROGRAMARI AMB UML

Origen de l'UML.

UML combina en una única notació els models següents:

- Modelatge de dades (semblant a un diagrama entitat-relació)
- Modelatge de fluxos d'entitats del sistema.
- Modelatge de fluxos d'un sistema.

UML no és una metodologia, tan sols és **una notació per a modelar el nostre sistema.**

DISSENY DE PROGRAMARI AMB UML

Utilitat d'UML

Un model és una abstracció d'un sistema o d'un problema que cal resoldre considerant un cert propòsit o un punt de vista determinat.

El model ha de descriure completament els aspectes rellevants del sistema.

Un diagrama ens permet representar gràficament un conjunt d'elements del model.

El que UML proposa és una notació i un conjunt de diagrames que inclouen les perspectives més rellevants del sistema.

DISSENY DE PROGRAMARI AMB UML

Utilitat d'UML

- .Existeixen molts tipus de diagrames UML. Tot i així, els més importants son:

. 1. **Diagrama de casos d'ús.**

. 2. **Diagrama de classes.**

. 3. Diagrames de comportament.
    a) Diagrama d'estats.
    b) Diagrama d'activitat.
    c) Diagrama d'interacció.
    d) **Diagrama de seqüència.**
    e) Diagrama de col·laboració.

. 4. Diagrames d'implementació.
    a) Diagrama de component..
    b) Diagrama de desplegament.

(durant aquesta UF estudiarem els diagrames que estan en negreta)

DISSENY DE PROGRAMARI AMB UML

Casos d'ús

Descriuen en forma d'accions el comportament del sistema estudiat des del punt de vista de l'usuari.

Exemple cas d'ús:
Considerem un sistema representat per una botiga de videojocs. La compra d'un videojoc és un cas d'ús per al sistema.
.

DISSENY DE PROGRAMARI AMB UML

Elements que intervenen als Casos d'ús

- Actor: És un usuari que interactua amb el sistema. La parella usuari-acció constitueix l'actor específic.

- Actor primari: És aquell per el qual el cas d'ús és essencial.

- Actor secundari: Interactuen amb el cas d'ús però el seu objectiu no és essencial.

DISSENY DE PROGRAMARI AMB UML

Casos d'ús

Exemple Actor:

- *Un comprador del videojoc és un actor primari per al cas d'ús que implica la compra.*
- *L'empresa o programador del videojoc així com el venedor de la botiga són actors secundaris per al cas d'ús que implica la compra.*

DISSENY DE PROGRAMARI AMB UML

Elements que intervenen als casos d'ús

Escenari

- De la mateixa forma que un objecte és una instància d'una classe, **un escenari és una 'instància' d'un cas d'ús**.

Exemple d'escenari:
La compra del videojoc 'Fifa 15' per part del client 'Pedro' és un exemple d'escenari per al cas d'ús 'Compra d'un videojoc'.

DISSENY DE PROGRAMARI AMB UML

Casos d'ús

Relació de comunicació: És la relació que vincula un actor amb un cas d'ús.

Aquesta relació inclou:

- Serveis i informació que l'actor pot introduir al sistema.

- Canvis que intervenen a l'entorn o dins del sistema.

DISSENY DE PROGRAMARI AMB UML

Casos d'ús

Exemple de relació de comunicació:

- *La compra del videojoc 'Fifa 15' per part del client Pedro genera una sèrie d'informació:*

- *tiquet de la compra.*

I pot haver subministrat d'altra informació:

- *tipus de pagament (efectiu, targeta)*
- *Dades de client (nom, cognom,...)*
-

DISSENY DE PROGRAMARI AMB UML

Diagrames de casos d'ús

- Representa la forma en què un actor i els elements del sistema interactuen entre sí.
- Són molt generals i intuïtius pel client

DISSENY DE PROGRAMARI AMB UML

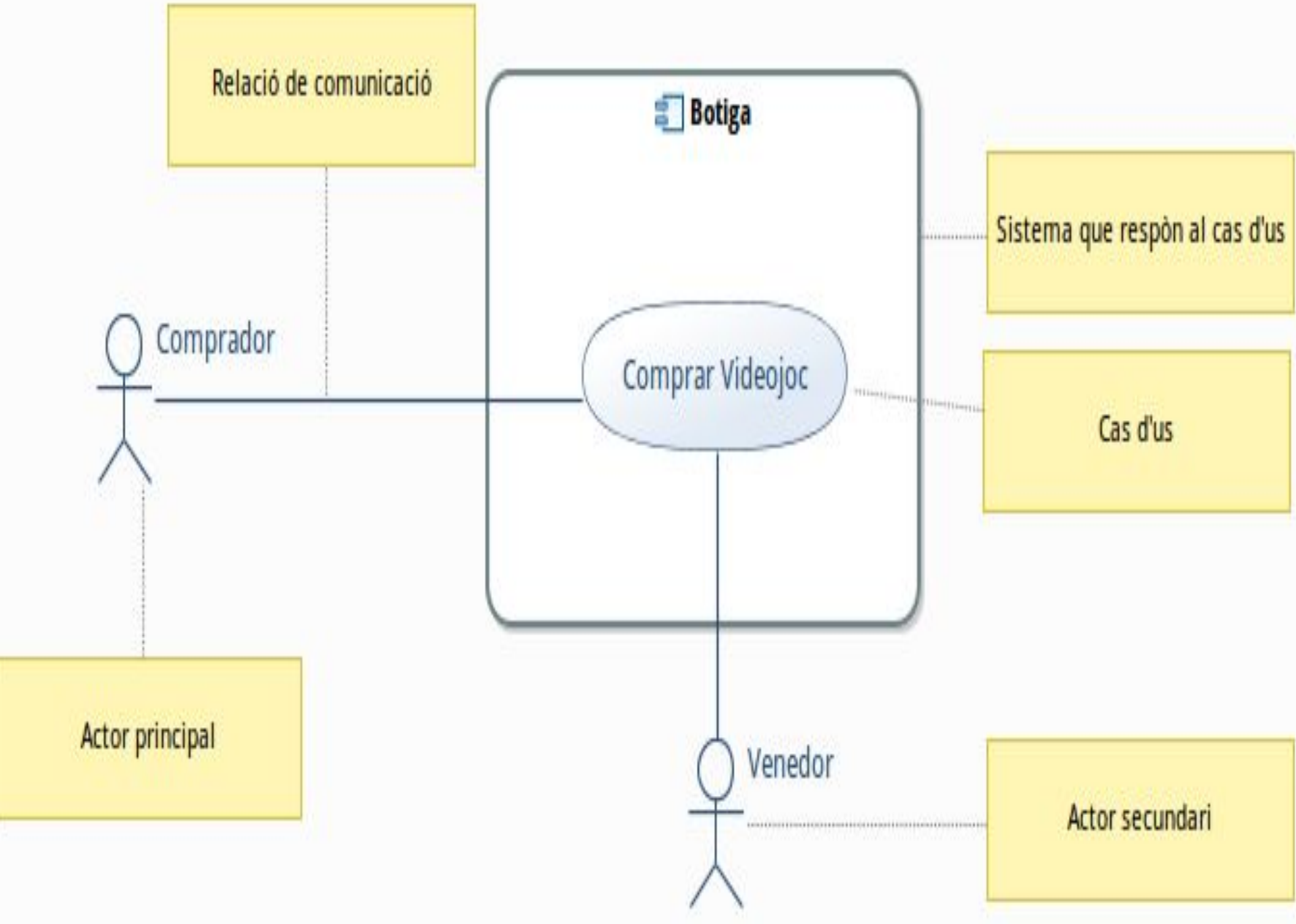
Diagrames de casos d'ús

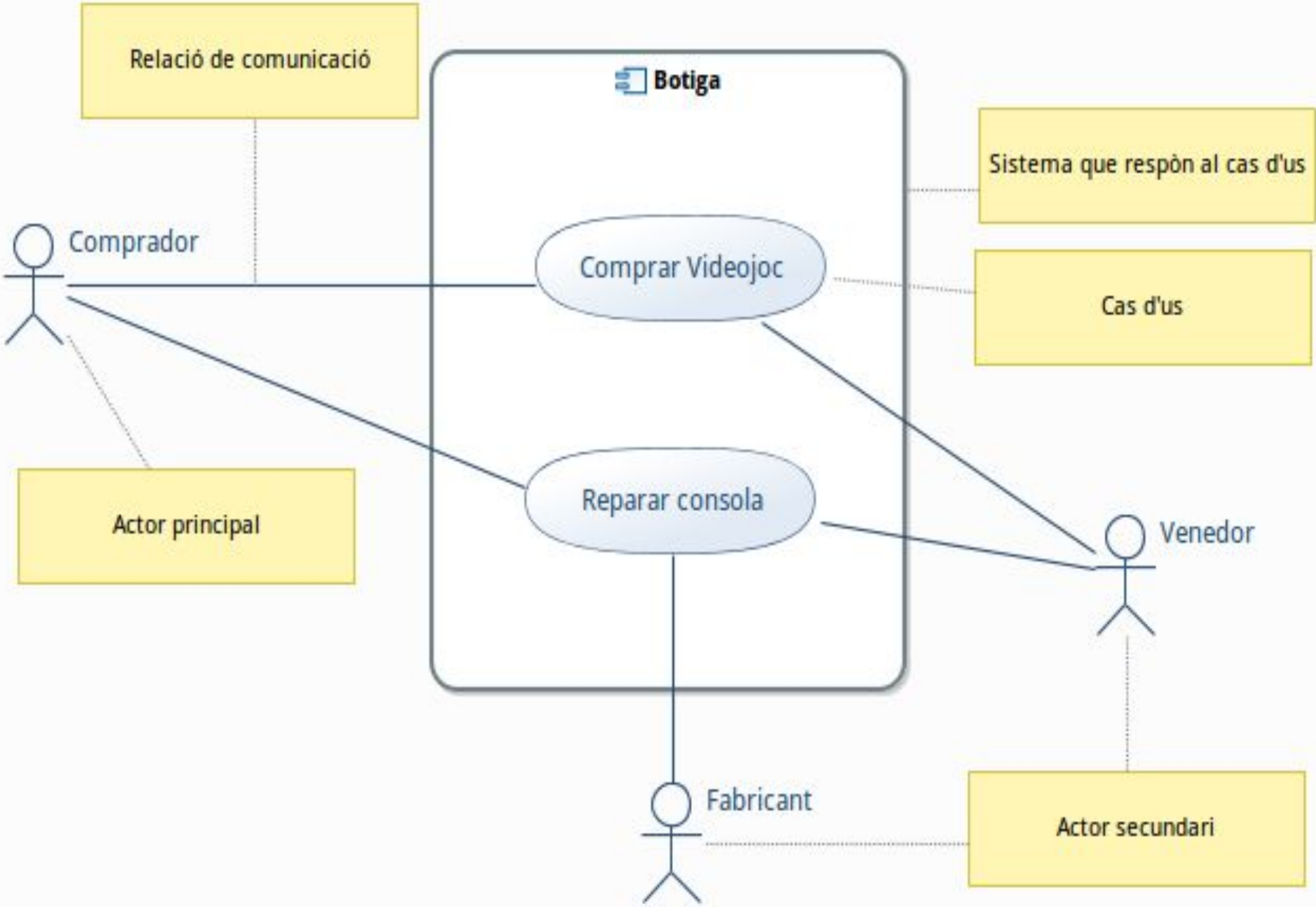
- Mostra els casos d'ús en forma d'el·lipse i els actors en forma de personatge.

- Les relacions de comunicació són línies que uneixen actors i casos d'ús.

- El sistema que respon a determinats casos d'ús es representa amb un rectangle que els conté.

(Vegeu la imatge a continuació…)


Relació de comunicació
Botiga
Sistema que respòn al cas d'us
Comprador
Comprar Videojoc
Cas d'us
Actor principal
Venedor
Actor secundari


Relació de comunicació
Botiga
Sistema que respòn al cas d'us
Comprador
Comprar Videojoc
Cas d'us
Actor principal
Reparar consola
Venedor
Fabricant
Actor secundari

DISSENY DE PROGRAMARI AMB UML

Diagrames de casos d'ús

- Els noms dels casos d'ús s'han de començar amb un verb.
- És més intuïtiu representar els casos d’ús i els actors en ordre descendent, situant els més importants a la part superior del diagrama.
- Anomena els actors amb substantius relacionats amb les regles de negoci, d'acord amb els rols que representen.

DISSENY DE PROGRAMARI AMB UML

Diagrames de casos d'ús

- Relació d'inclusió: Serveix per a enriquir un cas d'ús amb un altre.

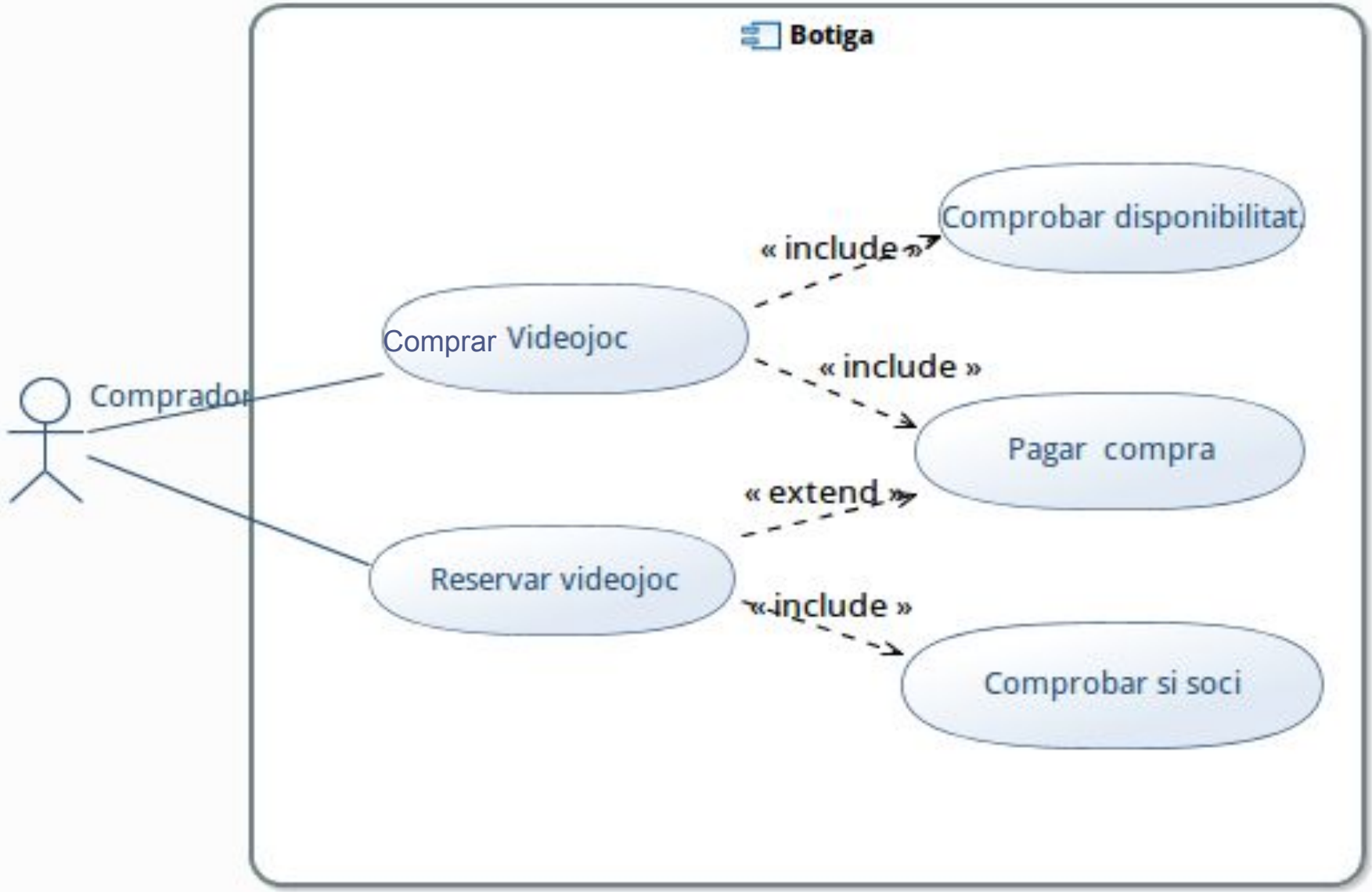
- El cas d'ús inclòs existeix únicament amb aquest propòsit, ja que no respon a un objectiu d'un actor primari.

- La inclusió és normalment una funcionalitat comú a varis casos d'ús.

- També pot servir per estructurar un cas d'ús en subcasos.

DISSENY DE PROGRAMARI AMB UML

.Diagrames de casos d'ús



DISSENY DE PROGRAMARI AMB UML

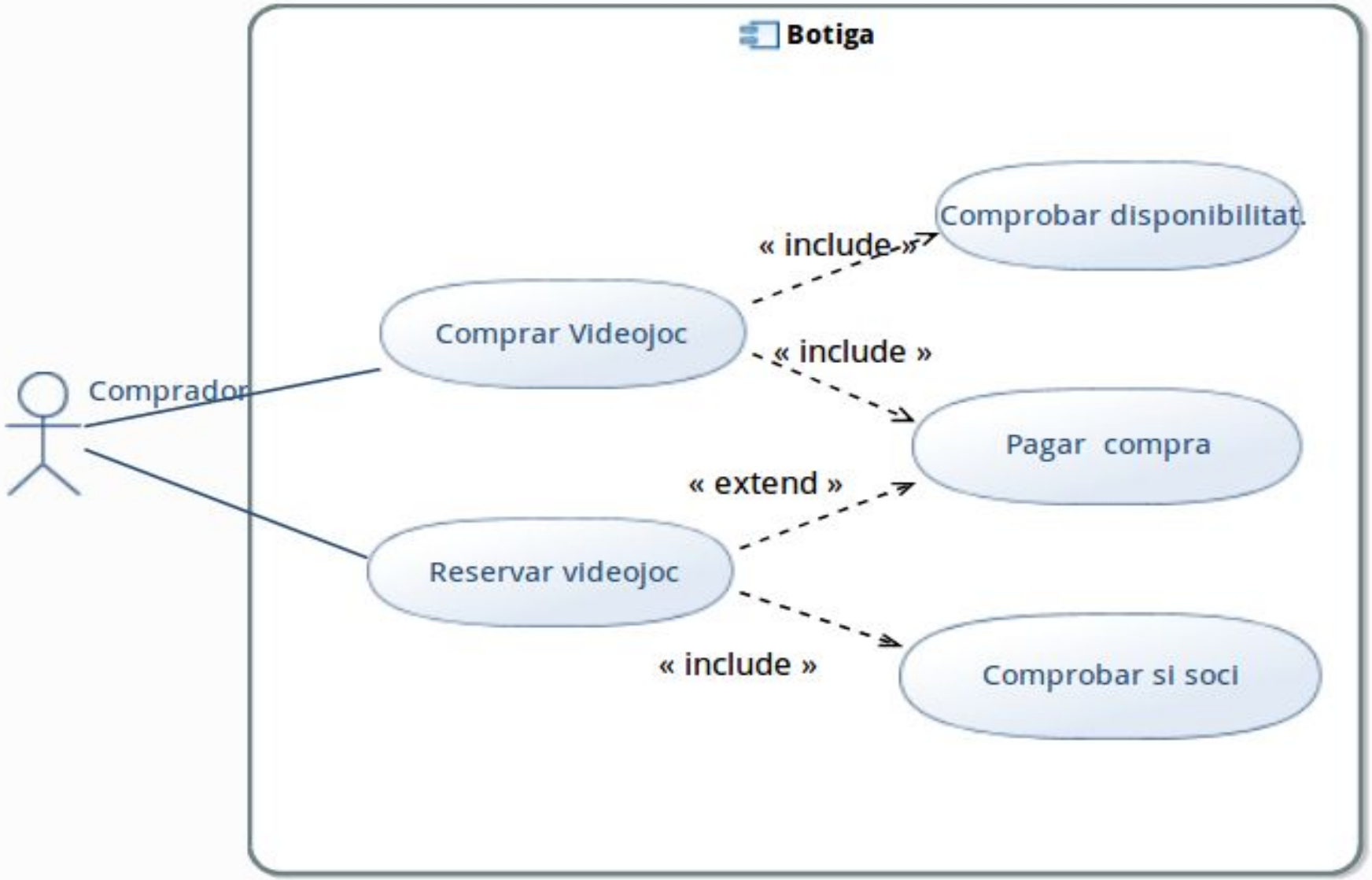
Diagrames de casos d'ús

- Relació d'extensió: És similar a la relació d'inclusió, tot i que, en aquest cas, és opcional.

- L'aplicació de l'extensió es decideix al desenvolupar un escenari del cas d'ús, remarcant així el seu caràcter opcional.

DISSENY DE PROGRAMARI AMB UML

.Diagrames de casos d'ús



DISSENY DE PROGRAMARI AMB UML

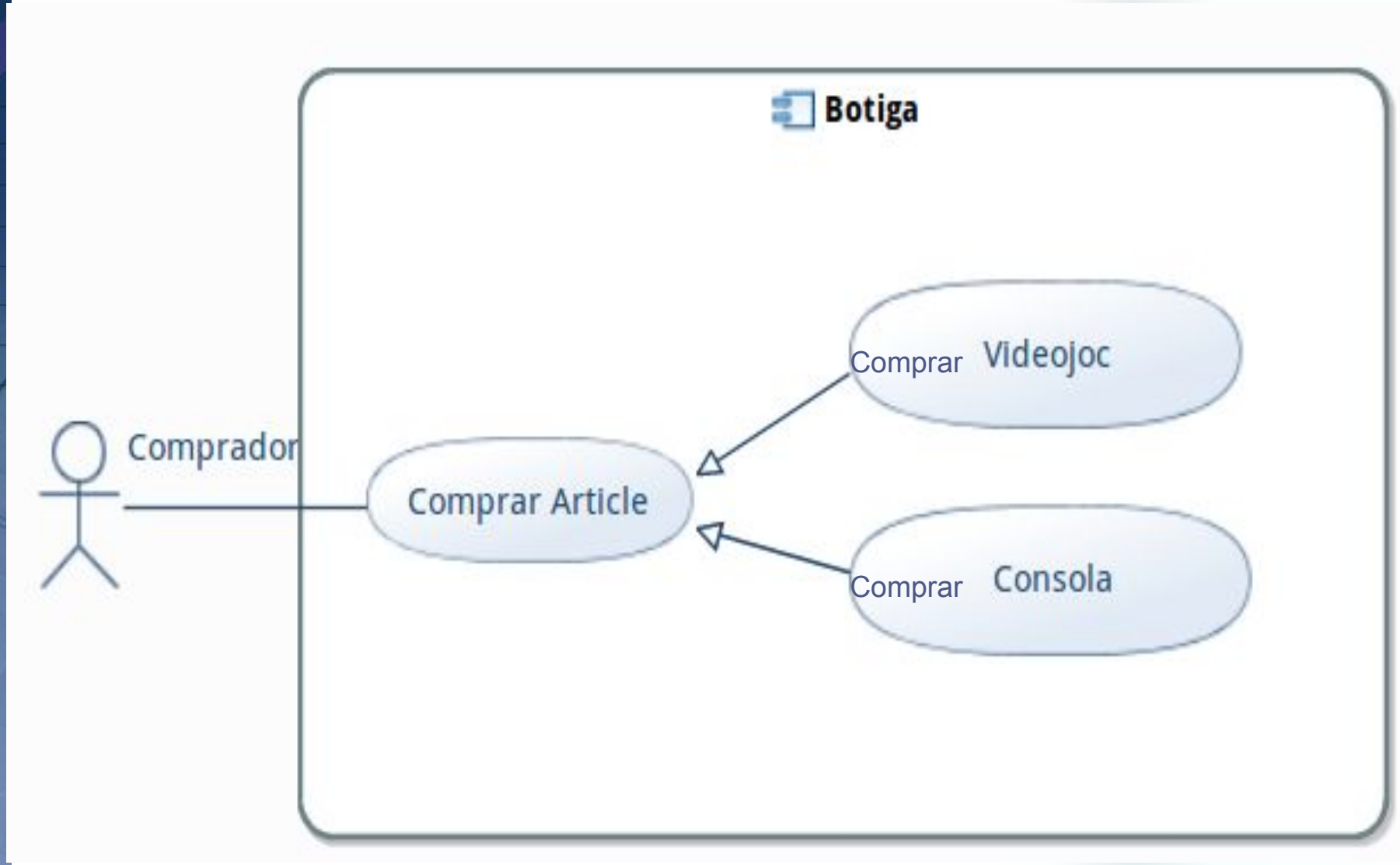
.Diagrames de casos d'ús

Generalització: Casos d'ús amb comportament similar es poden generalitzar en un cas d'ús que agrupi totes les seves característiques comunes.

Especialització: Les característiques pròpies dels casos d'ús esmentats anteriorment serien casos d'ús que 'derivarien' de la generalització.

DISSENY DE PROGRAMARI AMB UML

Diagrames de casos d'ús



DISSENY DE PROGRAMARI AMB UML

Creació de Diagrames de casos d'ús

Pasos

1. Llegir una vegada tot el text
2. Llegir una segona vegada i identificar:
   a. **Sistema**
   b. **Actors/rols**:
      i. Els principals a dalt a l'esquerra
      ii. Si hi ha rols de secundaris, a la dreta
3. Identificar les **accions/funcionalitats/casos d'ús**:
   a. Són verbs, accions
4. **Connectar els casos d'ús amb els actors**
   a. Hi ha herència d'actors?
5. Cal connectar casos d'ús entre ells? Amb **include o extend**
6. Llegir una última vegada per verificar i refer
7. Recordatori:
   a. No es representa cap seqüència o condició
   b. Poden existir múltiples solucions

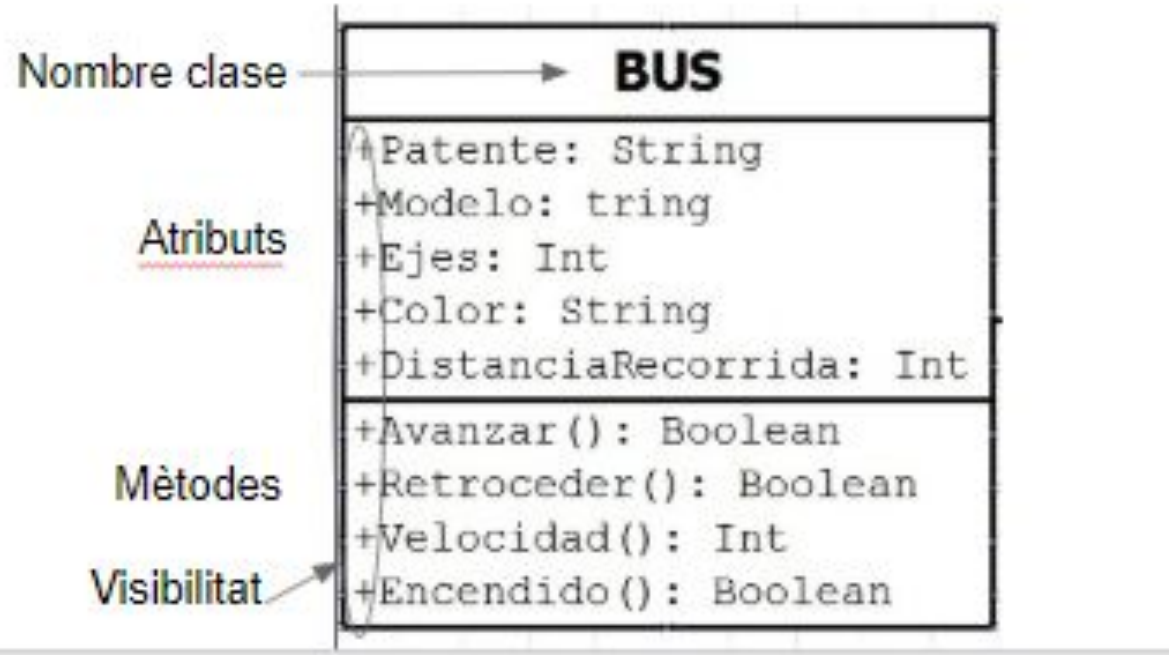
Diagrames de classe

DISSENY DE PROGRAMARI AMB UML

Diagrames de classe

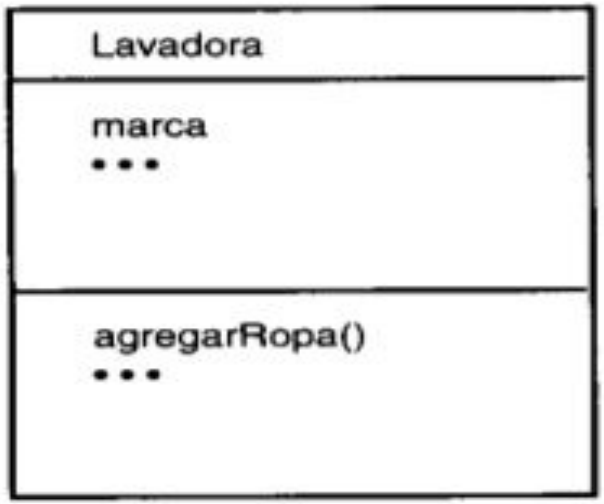
L'objectiu és representar les classes que formen el sistema i les seves relacions i dependències.

Una classe es representada en el diagrama de la següent manera:



DISSENY DE PROGRAMARI AMB UML

De manera simplificada una classe en el diagrama es pot representar així; enumerant els atributs i els mètodes:



I més simplificada encara:

DISSENY DE PROGRAMARI AMB UML

Diagrames de classe

Elements que intervenen al diagrama de classes:

- Classe: atributs, mètodes i visibilitat
- Relacions: d'associació, agregació, composició, herència i ús

DISSENY DE PROGRAMARI AMB UML

Diagrames de classe - Normes importants

- Recorda: **no existeix una solució única**
  - **Cal explicar la solució proposada SEMPRE**
  - **No us pagarán per fer un dibuix bonic, sinò per raonar**
- Normes:
  - Classes: substantiu, en singular, majúscula
  - Atributs: nomenclatura de Java, indicar tipus de dada i visibilitat
  - Mètodes: verb en infinitiu, nomenclatura de Java, indicant tipus de dada d'entrada, return i visibilitat
  - Relacions: verb, sempre indicar-la, nomenclatura de Java, posar sempre cardinalitat

DISSENY DE PROGRAMARI AMB UML

Diagrames de classe - Visibilitat (en atributs i mètodes)

- Public (+): Indica que serà visible tant dins com fora de la classe
- Private (-): Indica que només serà accessible des de dins de la classe
- Protected (#): Indica que només serà accessible des del package o les subclasses
- Per defecte(~), només visible pel package

DISSENY DE PROGRAMARI AMB UML

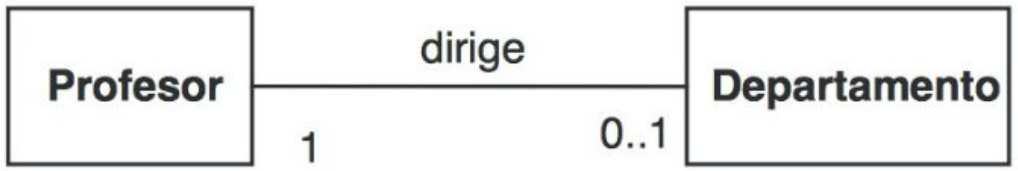
Diagrames de classe - Tipus de relacions entre classes

- Associació
- Dependència
- Herència
- S'ha de tenir en compte la multiplicitat a cada extrem de la relació:

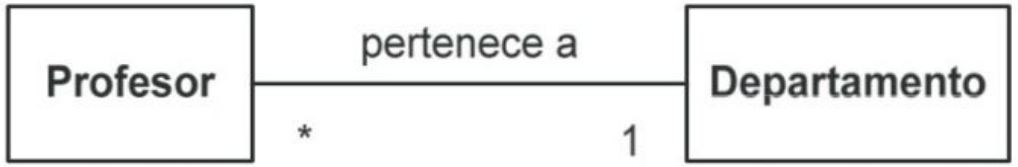
Multiplicidad	Significado
1	Uno y sólo uno
0..1	Cero o uno
N..M	Desde N hasta M
*	Cero o varios
0..*	Cero o varios
1..*	Uno o varios (al menos uno)

DISSENY DE PROGRAMARI AMB UML

Diagrames de classe - Relacions d'associació



Todo departamento tiene un director.
Un profesor puede dirigir un departamento.



Todo profesor pertenece a un departamento.
A un departamento pueden pertenecer varios profesores.

La relació d'associació es representa com una línia contínua entre dues classes, dirigida de la classe font a la classe destí.

Permet associar objectes que col·laboren entre sí.

DISSENY DE PROGRAMARI AMB UML

Diagrames de classe -

Relacions d'associació

Podem establir relacions d'associació reflexives.

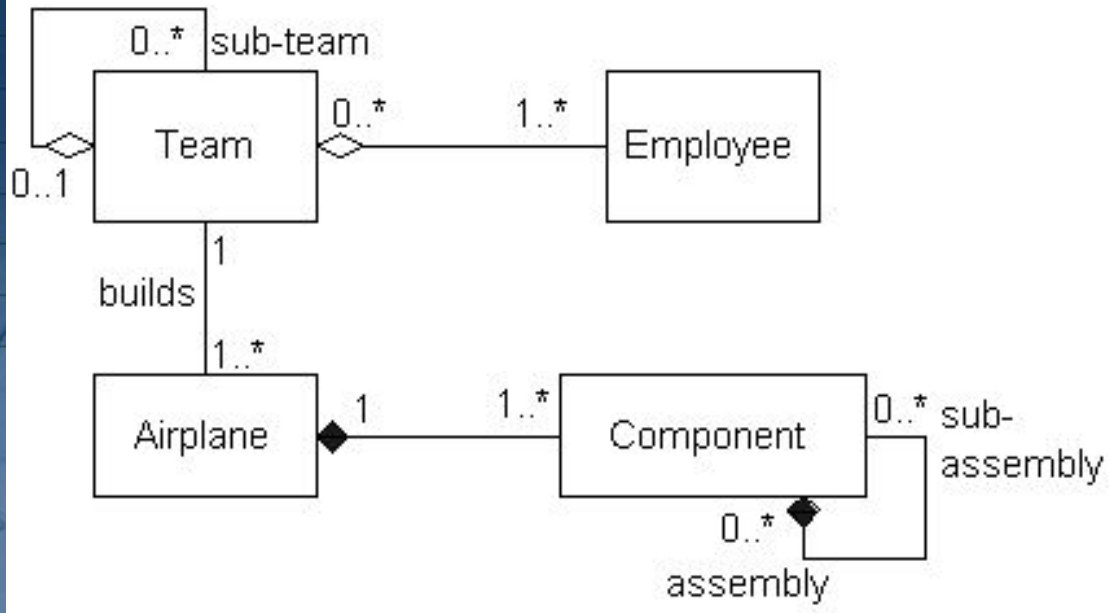
Una instància d'empleat pot dirigir a 0 o més empleats. Un empleat qualsevol pot ser dirigit per 0 o més empleats

Una persona es casada amb una altra persona. Si existeix la relació de casament entre dos persones, la cardinalitat és 1 als dos sentits

DISSENY DE PROGRAMARI AMB UML

Diagrames de classe -

Relacions d'Agregacions i composicions



AGREGACIÓ: Team - Employee: Un equip està format per 1 o més empleats. Si no hi ha equip, si pot haver-hi empleats.

COMPOSICIÖ: Airplane - Component. Un aeroplà està format per un o més components. Si en el sistema no hi ha aeroplans, no hi haurà tampoc components.

- Agregació: una classe forma part d'una altra (però **pot existir sense ella**)
- Composició: necessitem l'existència d'una classe per poder tenir una altra (**no té sentit l'existència d'una sense l'altra**)

DISSENY DE PROGRAMARI AMB UML

Diagrames de classe -

Relacions d'Agregacions i composicions

- L'agregació i la composició són dos tipus de relacions molt habituals que utilitzen els seus propis símbols en els diagrames.

- Les dues relacions s'utilitzen per indicar una inclusió dels objectes d'una classe en un objecte d'una altra classe.

- L'agregació representa una relació de tipus ***és-part-de***. Per exemple, els socis formen part d'un club, o uns clients formen part de la cartera de clients d'una empresa.

- La composició: és un tipus de relació que **implica exclusivitat.** Per exemple, un punt pot ser part d'un polígon o pot ser el centre d'un cercle, però no les dues coses a la vegada; o una agenda inclou les seves pàgines.

DISSENY DE PROGRAMARI AMB UML

Diagrames de classe

Composició vs Agregació de classes:

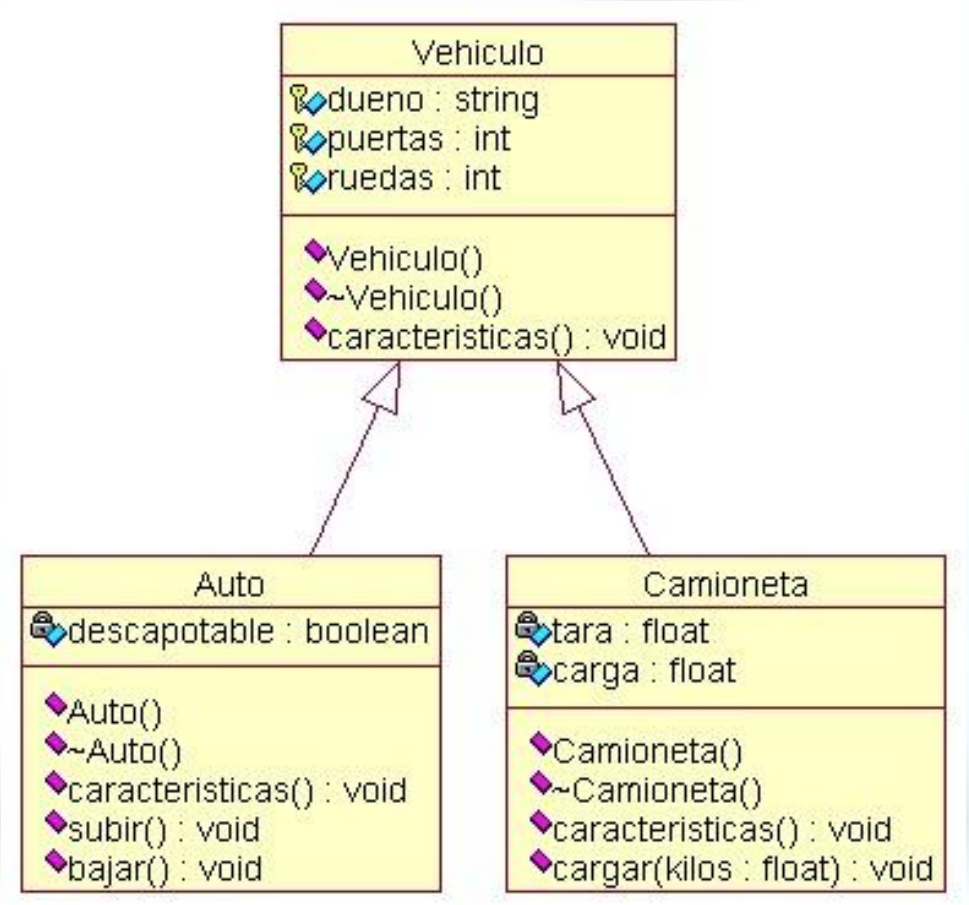
	Composició.	Agregació.
Varies associacions comparteixen els components.	No	Sí.
Es destrueixen els components quan es destrueix el compost.	Sí.	No.
Multiplicitat.	0..1 o 1.	Qualsevol.
Representació.	Diamant negre.	Diamant transparent.

Podeu consultar més informació sobre aquests tipus de relacions a
http://www.seas.es/blog/informatica/agregacion-vs-composicion-en-diagramas-de-clases-uml/

DISSENY DE PROGRAMARI AMB UML

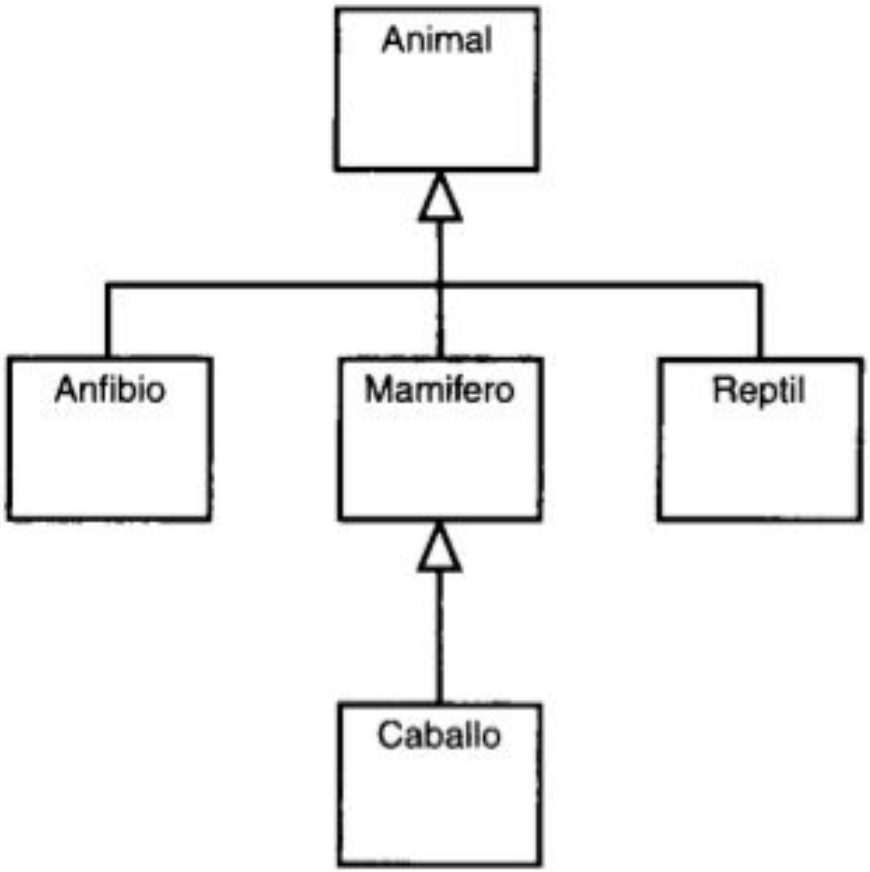
Diagrames de classe
Relacions d'herència

- Una superclasse abstracta
- Una subclasse que hereta els atributs i mètodes **pùblics i protected** de su superclasse



Diagrames de classe - Herència

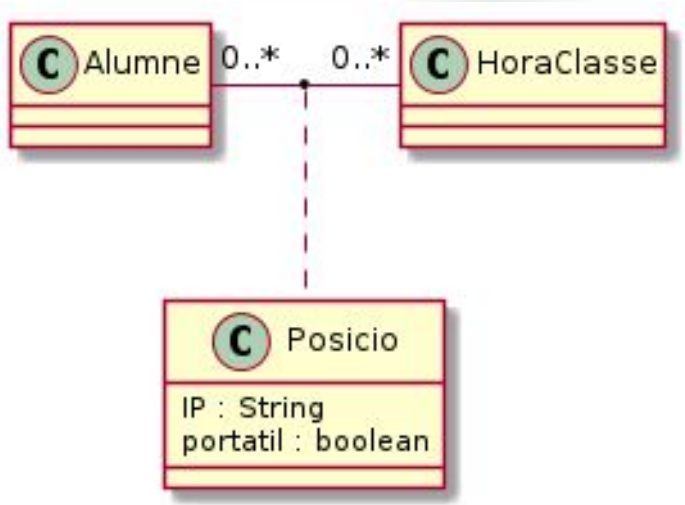
Altre exemple:



DISSENY DE PROGRAMARI AMB UML

Diagrames de classe
Classes associatives

- Una classe associativa ens permet establir propietats i mètodes en la relació d'associació entre dues classes.
- A l'exemple, volem representar la relació de reserva de lloc de cada alumne en una hora de classe concreta.
- Amb la classe associativa Posició podem expressar que només hi pot haver una posició per a cada parella d'alumne-classe

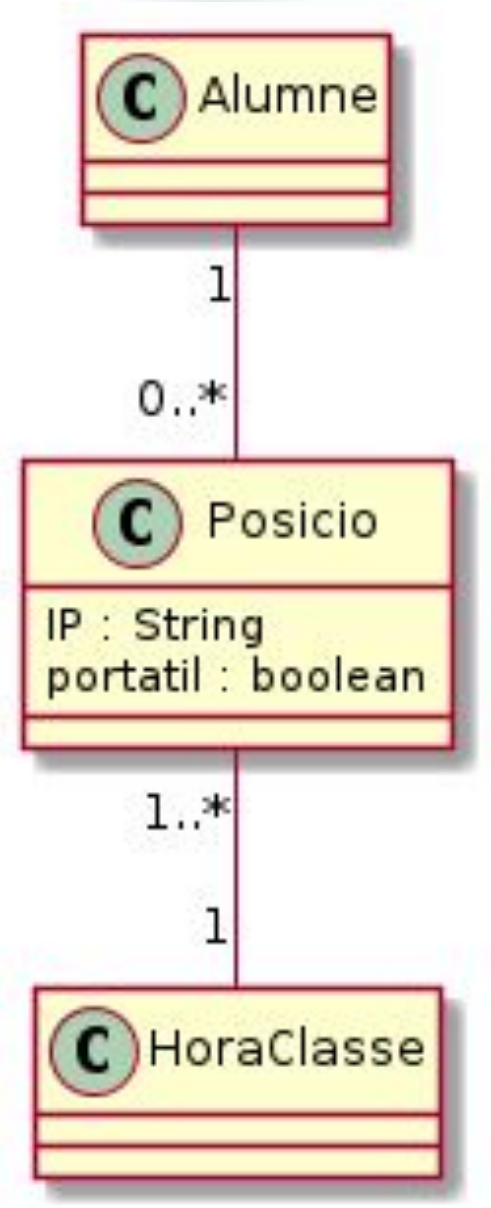


DISSENY DE PROGRAMARI AMB UML

El mateix cas sense Classes associatives

- En aquest cas, res impedeix que creem dues posicions diferents per al mateix alumne a la mateixa hora de classe.

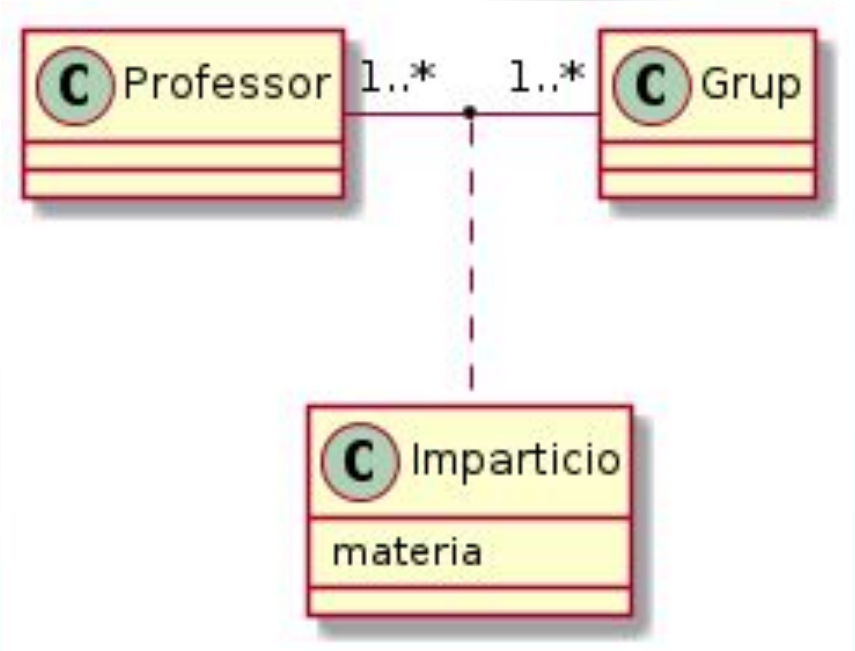
- Així, **una classe associativa afegeix una restricció que fa referència a la relació entre les dues classes que enllaça.**



DISSENY DE PROGRAMARI AMB UML

Classes associatives - Altre exemple

- Fixeu-vos que segons aquest diagrama no és possible que un professor imparteixi dues matèries diferents al mateix grup.
- Si aquesta és una possibilitat en el nostre model, la classe Impartició no hauria de ser una classe associativa.
- La implementació d'una classe associativa es pot fer simplement com si fos una classe habitual, amb mètodes que permetin accedir als objectes associats.



Diagrames de seqüència

DISSENY DE PROGRAMARI AMB UML

Diagrames de seqüència

- El diagrama de seqüència descriu la dinàmica del sistema.

- Si el diagrama de classe representa el comportament estàtic del sistema el diagrama de seqüència representa el comportament dinàmic del sistema.

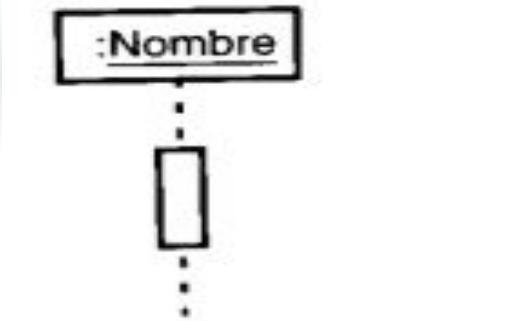
- Dit d'una altra forma el diagrama de seqüència es una representació gràfica que mostra:
  - Com funcionen les classes internament.
  - Com interactúen les classes entre elles.

DISSENY DE PROGRAMARI AMB UML

Diagrames de seqüència

El diagrama de seqüència consta d'objectes que es representen de la següent forma:

- Rectangles amb nom.
- Missatges representats amb línies continues amb punta de fletxa.
- El temps, representat com a una progressió vertical.



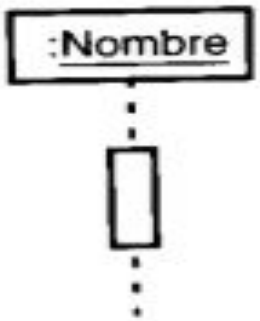
DISSENY DE PROGRAMARI AMB UML

Diagrames de seqüència

Els objectes es col·loquen a la part superior del diagrama, d'esquerra a dreta.

- L'extensió vertical de l'objecte rep el nom de línia de vida.
- El petit rectangle a la línia de vida representa l'activació de l'objecte.

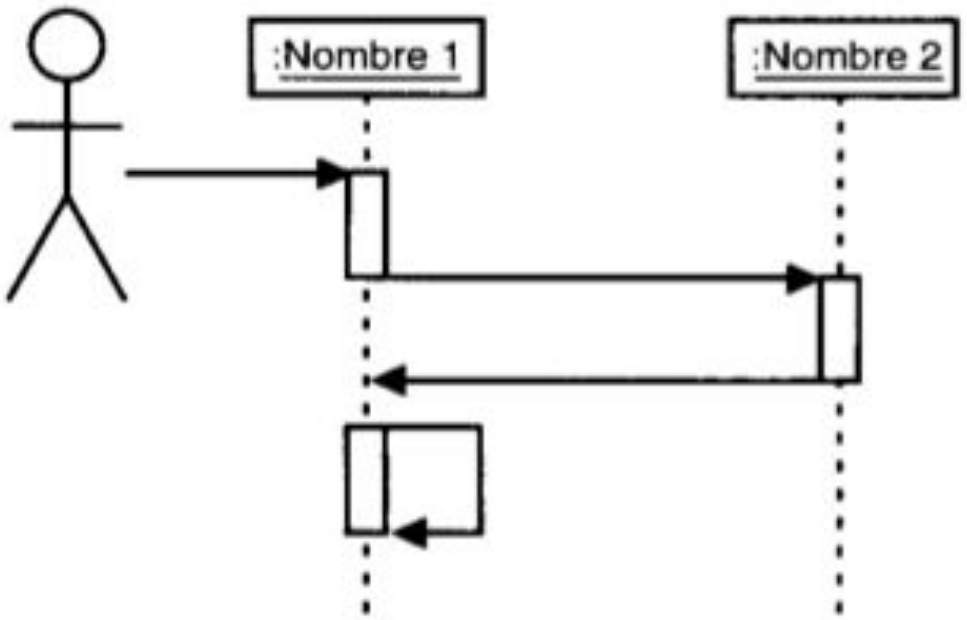
La longitud del rectangle es correspon amb la duració de l'activació.



DISSENY DE PROGRAMARI AMB UML

Diagrames de seqüència

- Un missatge va d'un objecte a un altre i es representa per una fletxa. Un objecte es pot enviar missatges a si mateix.
-



DISSENY DE PROGRAMARI AMB UML

Diagrames de seqüència

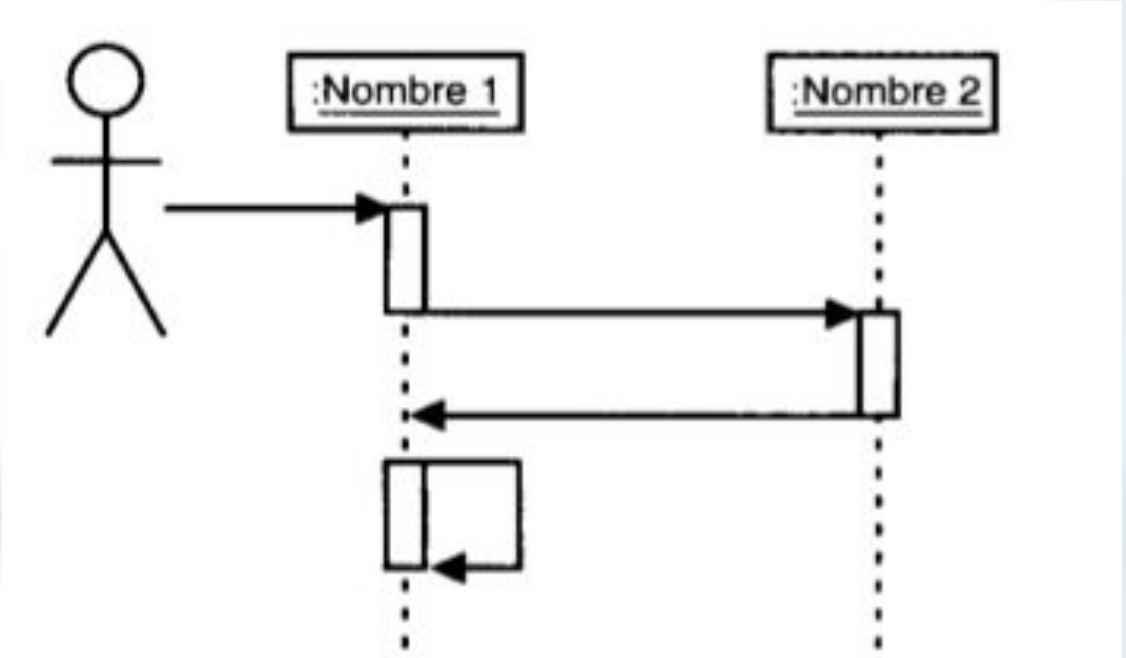
- Un missatge pot ser:
- 
-
-
-
- Síncron: L'objecte que envia espera la resposta del destí abans de continuar amb la seva feina.

- Asíncron: L'objecte que envia no espera resposta abans de continuar.

DISSENY DE PROGRAMARI AMB UML

Diagrames de seqüència

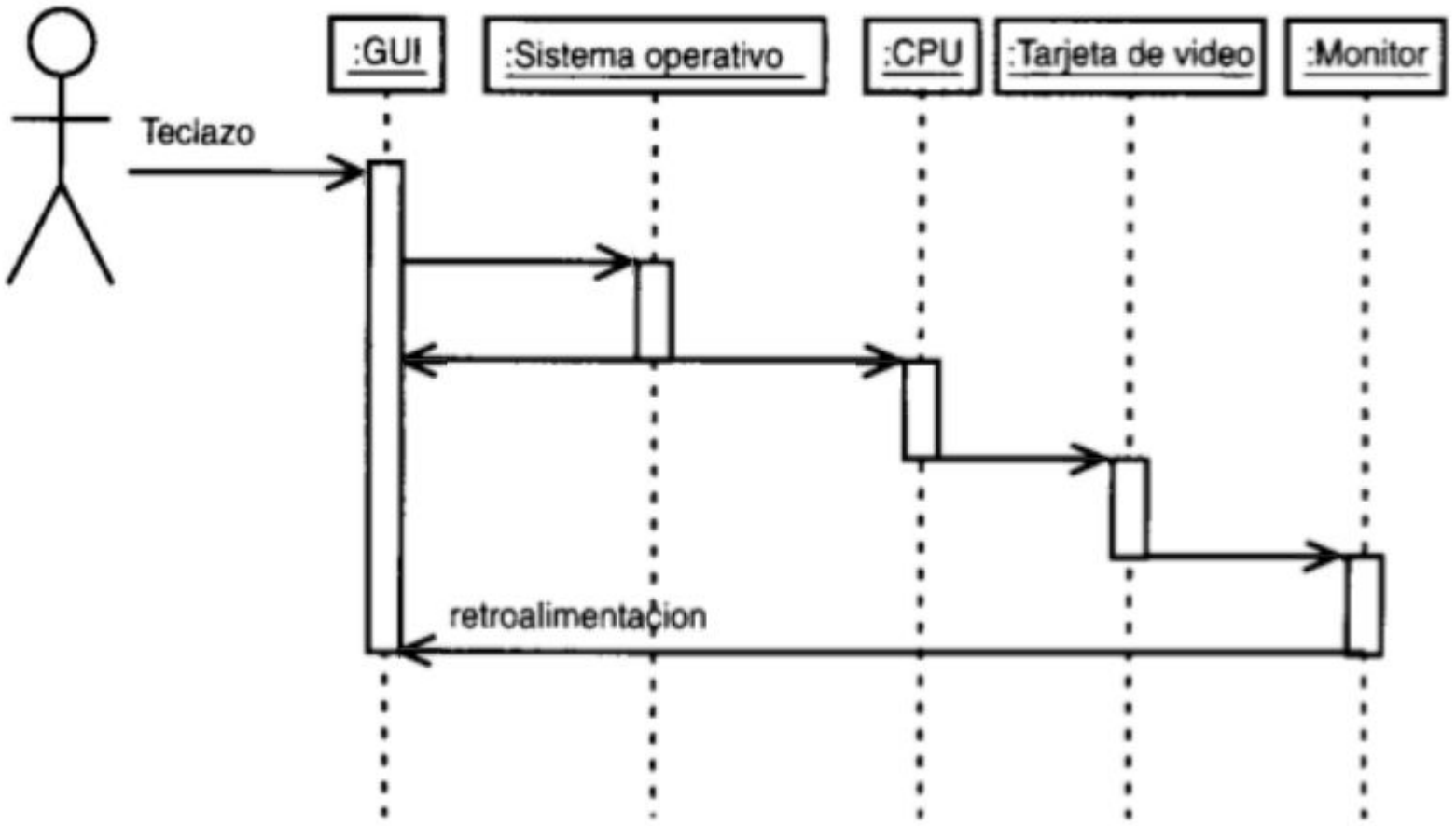
- Com hem dit, el temps es representa verticalment. S'inicia a la part superior i avança cap a la inferior.
-



DISSENY DE PROGRAMARI AMB UML

Diagrames de seqüència

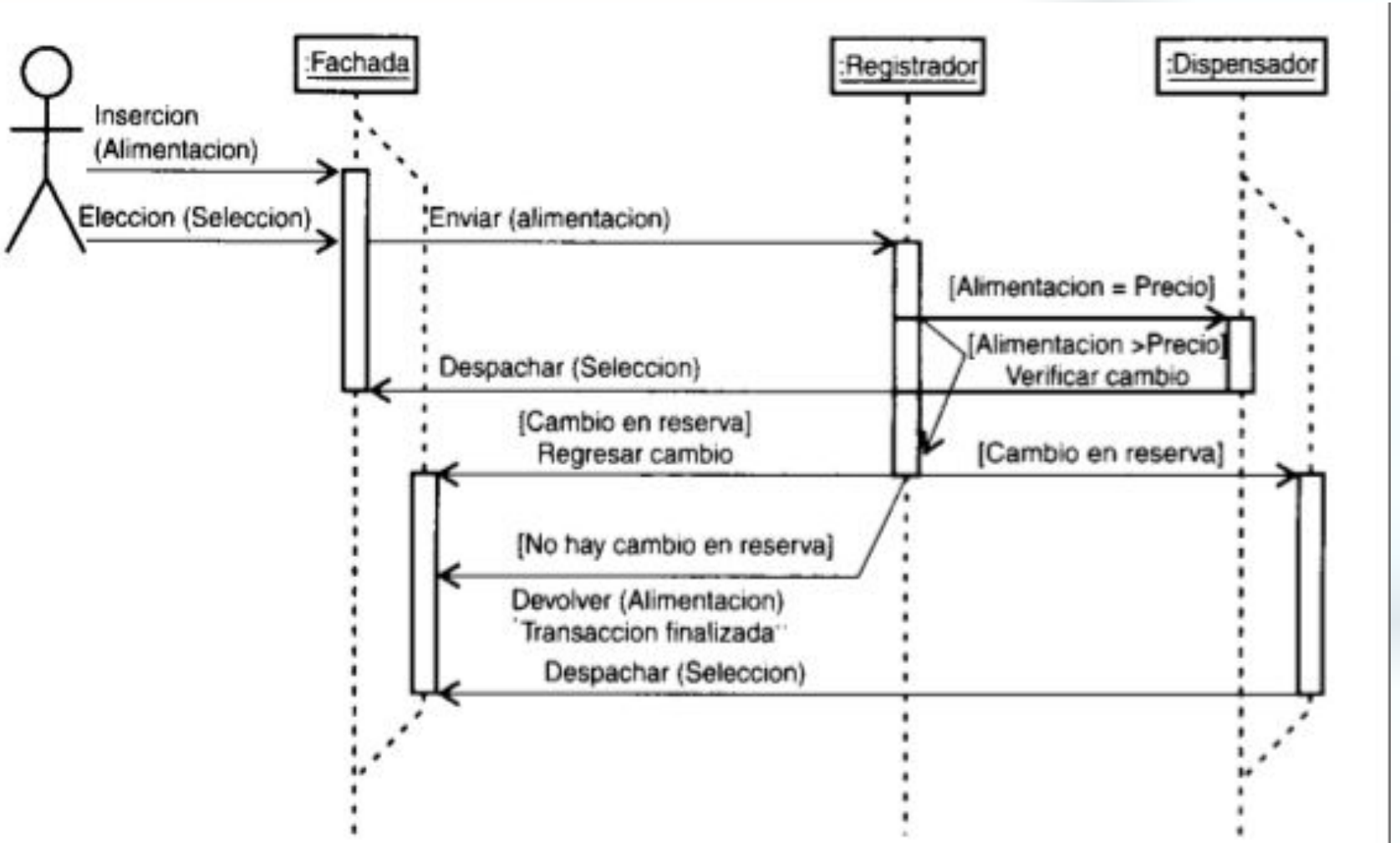
- Exemple 1: Un usuari presiona una tecla a una interfície gràfica d'un processador de text:



DISSENY DE PROGRAMARI AMB UML

Diagrames de seqüència

•Exemple 2: Dispensador de begudes amb control d'import incorrecte.



DISSENY DE PROGRAMARI AMB UML

Diagrames de seqüència

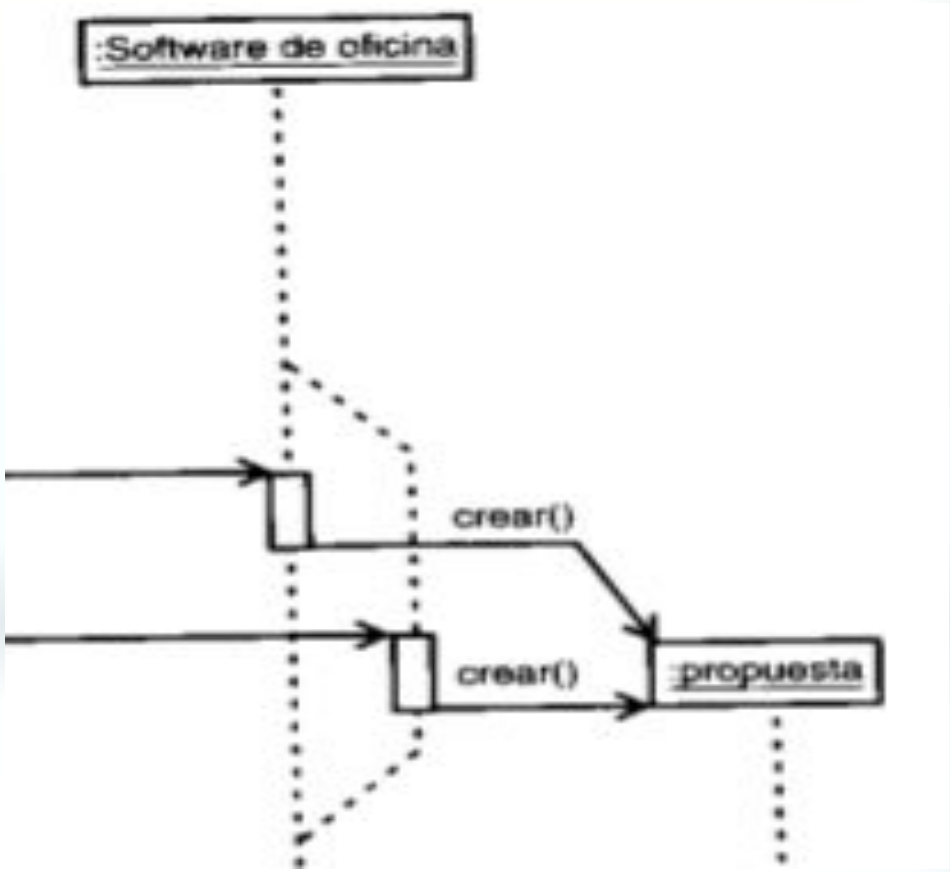
- Si el cas anterior incorpora tots els casos d'ús al diagrama de seqüència, es coneix com a diagrama de seqüència genèric.

- Un altre punt important és com representar a un diagrama de seqüència la creació, en temps d'execució, d'un objecte instanciat a partir d'una classe ja definida.

DISSENY DE PROGRAMARI AMB UML

Diagrames de seqüència

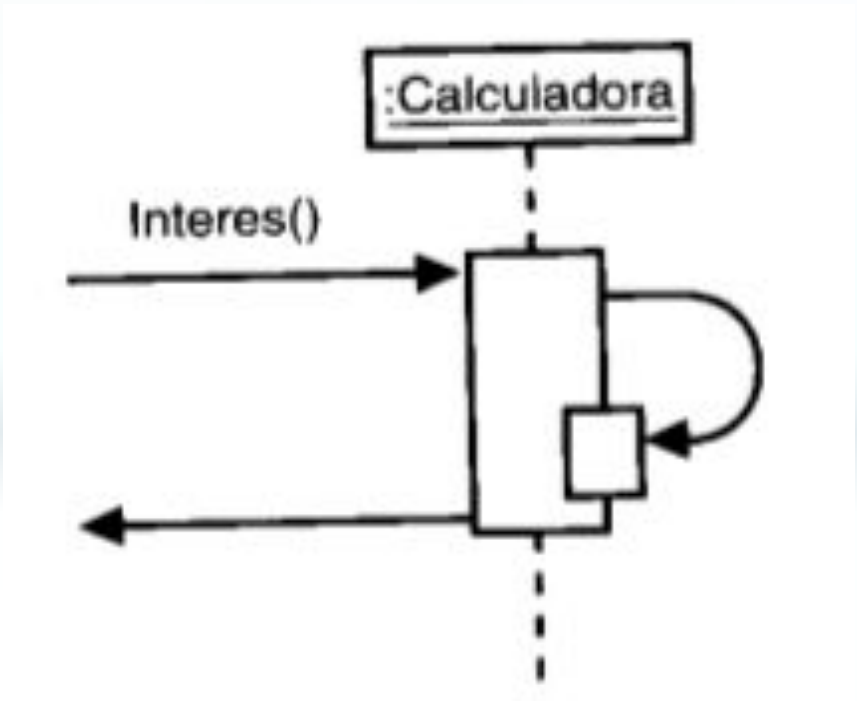
- El següent exemple mostra aquesta possibilitat.
- Observeu que el nou objecte es crea una mica més a sota de la resta d'objectes acompanyat del missatge crear().



DISSENY DE PROGRAMARI AMB UML

Diagrames de seqüència

- Per representar recursivitat en un objecte o una crida a un mètode del propi objecte que només afecta a aquest es representarà així:



DISSENY DE PROGRAMARI AMB UML

Diagrames de seqüència

Exemple 3: préstec d'un llibre. Mostrem la seqüència de interaccions entre classes.
Posem:
.Classes
.Mètodes
.Returns