

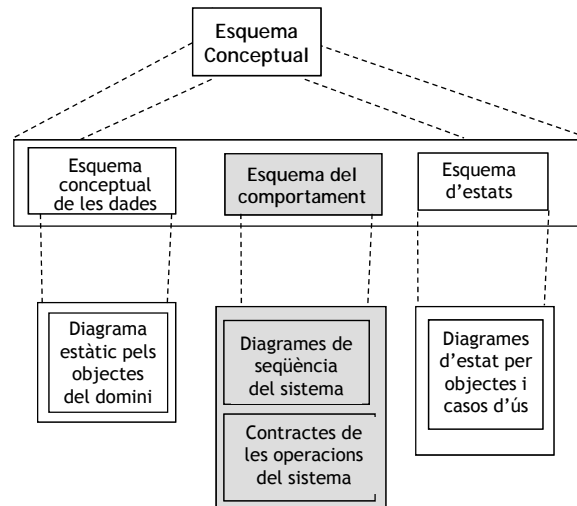
Especificació en UML: Esquema del comportament



Esquema del comportament en UML

- Especificació del comportament 'orientat a objectes'
- Diagrames de seqüència del sistema
- Contractes de les operacions del sistema
- Redundància amb l'esquema conceptual de les dades
- Interacció amb actors software (serveis) externs
- Especificació en OCL de la sortida d'una operació
- Diagrama d'estats
- Bibliografia

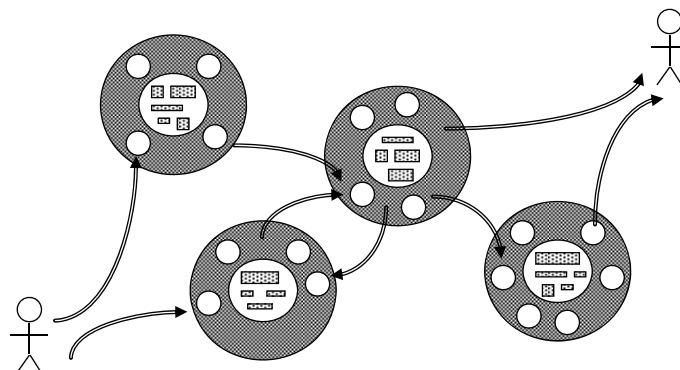
Esquema conceptual (especificació)



3

Descripció del comportament en OO (visió dinàmica d'un sistema OO)

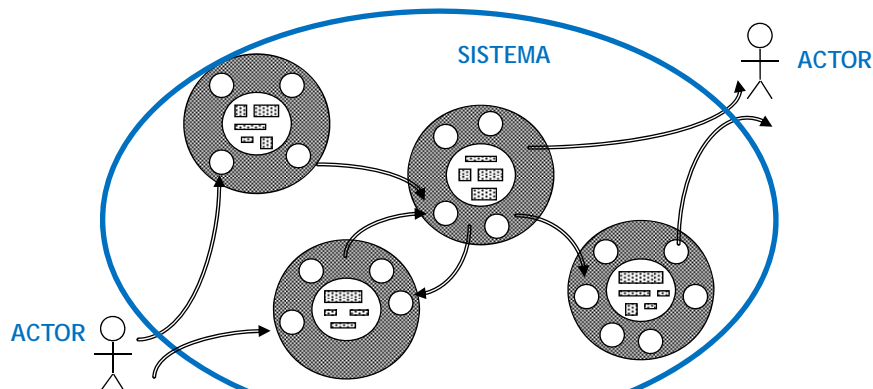
Els objectes es comuniquen (interactuen) mitjançant la invocació d'operacions d'altres objectes



4

“Especificació” del comportament en OO

- Considerem un tipus especial “sistema” que engloba tots els objectes i describem la interacció dels actors externs amb el sistema
- L'esquema del comportament del “sistema” ens permet especificar la dinàmica dels objectes descrits al *diagrama de classes* i dels *escenaris dels casos d'ús*.



5

Artefactes usats a l'esquema del comportament del sistema

Diagrames de seqüència del sistema (DSS): Mostra les interaccions entre els actors i el sistema.

- Punt de partida: La *seqüència d'esdeveniments en un escenari d'un cas d'ús*
- Idea bàsica: L'actor genera esdeveniments cap al sistema, que exigeixen l'execució d'alguna operació com a resposta.
- Objectiu/utilitat: Permeten identificar les operacions del sistema i especificar la dinàmica dels escenaris.

Contractes de les operacions del sistema: Descriuen l'efecte de les operacions del sistema

- Punt de partida: Operacions identificades del sistema
- Idea bàsica: especificació Pre/Post per descriure els canvis d'estat que es produeixen com a conseqüència de l'execució
- Objectiu/utilitat: Permeten establir/fixar el comportament desitjat de les operacions

6

Només per veure l'aspecte dels DSS i els contractes:



Cas d'ús: Afegir-Persona

1. La persona indica al sistema el seu nom i la seva adreça
2. El Sistema enregistra el nom i l'adreça de la persona
3. La persona indica al sistema l'idioma que parla
4. El Sistema enregistra l'idioma parlat per la persona

Contractes:

Operació: `altaPersona(nom,adreça)`

Precondicions:

Postcondicions:

- s'ha creat d'un objecte Persona amb el nom i l'adreça especificats

Sortida:

Operació: `parlaldioma(nom,nomId)`

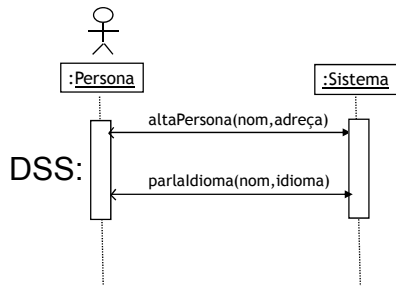
Precondicions:

- l'idioma identificat per nomId existeix

Postcondicions:

- creació d'una associació Parla entre la persona amb nom=nom i l'idioma nom=nomId

Sortida:



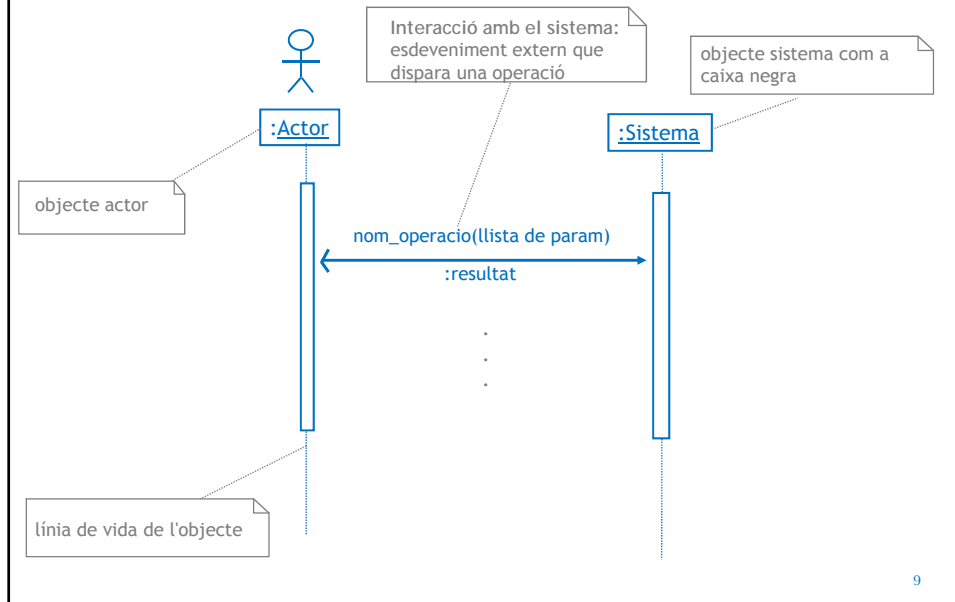
7

Avui anem a veure:

- Elements dels Diagrames de seqüència del sistema
- Elements dels Contractes de les operacions del sistema
- Redundància amb l'esquema conceptual de les dades

8

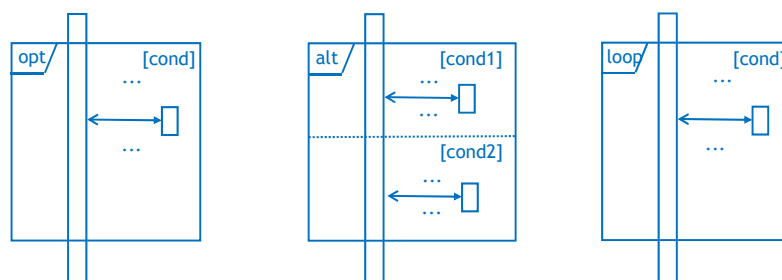
Elements bàsics d'un DSS:



9

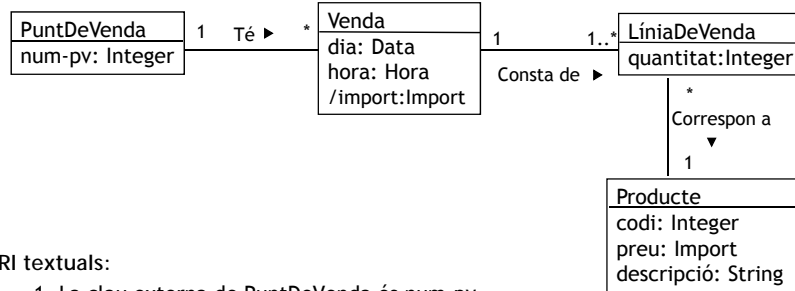
Elements d'un DSS: frames

- Els *frames* permeten estructurar informació en els diagrames UML
- Tres tipus principals de *frames*:
 - execució opcional (*opt*)
 - execució alternativa (*alt*)
 - execució repetida (*loop*)
- Els *frames* es poden aniar lliurement



10

Recordem l'Exemple del caixers i els punt de venda:



RI textuais:

- 1- La clau externa de PuntDeVenda és num-pv
- 2- La clau externa de Producte és codi
- 3- Un punt de venda no pot tenir més d'una venda amb el mateix dia i hora
(per tant una instància de venda queda unívocament determinada per tres valors de num-pv, dia i hora)

Informació derivada:

- 1- L'import d'una venda és igual al cost total dels productes de que consta

11

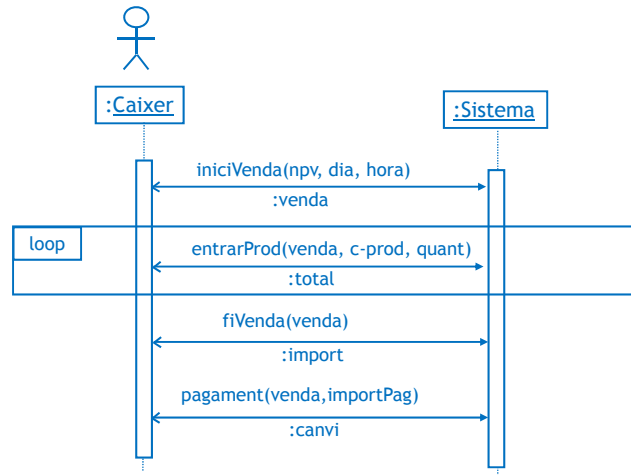
Exemple: cas d'ús Comprar Productes

escenari principal d'èxit

1. El Caixer indica al Sistema que comença una nova venda, amb el dia i l'hora actuals, en una caixa que té un número determinat
2. El Sistema enregistra el començament de la venda a la caixa
3. El Caixer entra al Sistema l'identificador de producte i la quantitat d'unitats
4. El Sistema enregistra la línia de venda corresponent al producte
5. El Sistema mostra el total acumulat
Es repeteixen els passos 3 a 5 fins que s'han processat tots els productes
6. El Caixer indica al Sistema l'acabament de la venda
7. El Sistema presenta el total de la venda
8. El Caixer comunica al Client el total de la venda
9. El Client entrega al Caixer una quantitat de diners, potser més gran que el total de la venda
10. El Caixer indica al Sistema els diners que ha rebut
11. El Sistema mostra el canvi i imprimeix el rebut
12. El Caixer diposita els diners rebuts a la caixa i n'extreu el canvi
13. El Caixer dóna el canvi al Client
14. El Client se'n va amb els productes comprats

12

Exemple: Comprar Productes



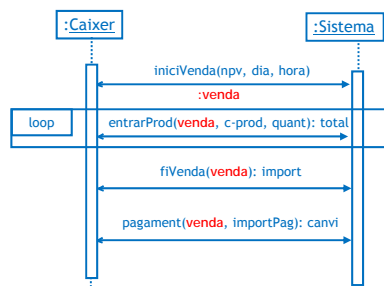
venda és un objecte del sistema que correspon a la venda que s'està realitzant
El valor d'aquest paràmetre és el mateix a totes les invocacions

13

Compartició d'informació entre operacions d'un DSS

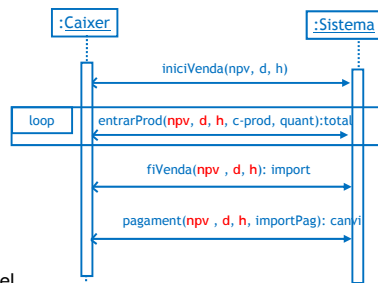
UML no precisa de quina manera les operacions d'un diagrama de seqüència del sistema poden compartir informació

Nosaltres ho especificarem mitjançant arguments addicionals de les operacions



Restricció implícita: el valor de l'argument 'venda' és el mateix a tots els esdeveniments del diagrama

Alternativament, ús de la clau externa:



Restricció implícita: el valor dels argument npv, d i h és el mateix a tots els esdeveniments del diagrama

14

Recordem:

Diagrames de seqüència del sistema (DSS): Mostra les interaccions entre els actors i el sistema.

- Punt de partida: La *seqüència d'esdeveniments en un escenari d'un cas d'us*
- Idea bàsica: L'actor genera esdeveniments cap al sistema, que exigeixen l'execució d'alguna operació com a resposta.
- Objectiu/utilitat: Permeten identificar les operacions del sistema i especificar la dinàmica dels escenaris.

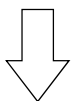
Contractes de les operacions del sistema: Descriuen l'efecte de les operacions del sistema

- Punt de partida: Operacions identificades del sistema
- Idea bàsica: especificació Pre/Post per descriure els canvis d'estat que es produeixen com a conseqüència de l'execució
- Objectiu/utilitat: Permeten establir/fixar el comportament desitjat de les operacions

15

Contractes: Esdeveniments i operacions

- Esdeveniment extern generat per un actor



La comunicació d'un esdeveniment del sistema provoca l'execució d'una operació del sistema amb el mateix nom i els mateixos paràmetres

- Operació del sistema: Operació interna que s'executa com a resposta a la comunicació de l'esdeveniment

16

Operacions del sistema

NO s'assignen a objectes concrets durant l'etapa d'especificació

Les operacions del sistema s'agrupen com operacions de la classe "Sistema", la qual usem per a representar el sistema software.

Exemple:

Sistema
iniciVenda(npv: Integer, d:dia, h:hora): venda
entrarProd(venda:Venda, c-prod: Integer, quant: Integer): total
fiVenda(venda: Venda): Import
pagament(venda: Venda, importPag: Import): Import

17

Elements dels contractes de les operacions

Operació: nom i paràmetres de l'operació (Signatura de l'operació)

Precondicions:

Condicions que cal satisfer per poder executar l'operació.

Postcondicions:

Canvis d'estat que es produeixen com a conseqüència de l'execució:

- altes/baixes d'instàncies de classes d'objectes i d'associacions
- modificació d'atributs
- generalització o especialització d'un objecte
- canvi de subclasse d'un objecte

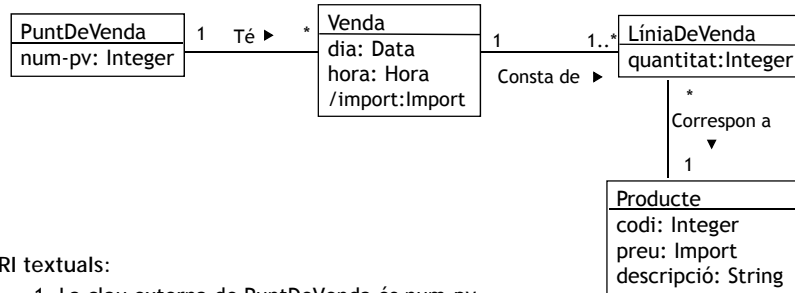
Sortida:

Descripció de la sortida que proporciona l'operació

Observeu el lligam que existeix entre els contractes de les operacions i l'esquema conceptual de les dades

18

Exemple: esquema conceptual de partida



RI textuais:

- 1- La clau externa de PuntDeVenda és num-pv
- 2- La clau externa de Producte és codi
- 3- Un punt de venda no pot tenir més d'una venda amb el mateix dia i hora
(per tant una instància de venda queda unívocament determinada per tres valors de num-pv, dia i hora)

Informació derivada:

- 1- L'import d'una venda és igual al cost total dels productes de que consta

19

Exemple: operació *iniciVenda*

Operació: *iniciVenda*(npv: Integer, dia:Data, hora:Hora): Venda

Precondicions:
Existeix un PuntDeVenda amb num-pv=npv

Postcondicions:
S'ha creat una instància V de Venda amb dia i hora
V s'ha associat amb la instància de PuntDeVenda amb num-pv=npv

Sortida: V

En OCL:

context: Sistema::*iniciVenda*(npv:Integer, dia:Data, hora:Hora): Venda

pre: PuntDeVenda.allInstances()->exists(p|p.num-pv = npv)

post:
Venda.allInstances()->exists(v|v.oclIsNew() and v.dia=dia
and v.hora=hora and v.puntDeVenda.num-pv=npv and result =v)

20

Exemple: operació *entrarProd*

Operació: `entrarProd(venda: Venda, c-prod: Integer, quant: Integer): Import`

Precondicions:

Existeix un *Producte* amb `codi = c-prod`

Postcondicions:

S'ha creat una instància de *LíniaDeVenda* *L* amb `quantitat=quant`

venda s'ha associat amb *L*

L s'ha associat amb el *Producte* amb `codi = c-prod`

Sortida:

Es retorna l'import acumulat de la venda

En OCL:

context: `Sistema::entrarProd(venda:Venda, codi:Integer, quant:Integer)`

pre: `Producte.allInstances()->exists(p|p.codi = codi)`

post:

`LíniaDeVenda.allInstances()->exists(l|l.oclIsNew() and
l.quantitat = quant and l.venda = venda
and l.producte.codi = codi and result=venda.import)`

21

Exemple: operació *fiVenda*

Operació: `fiVenda(venda: Venda) :Import`

Precondicions:

Postcondicions:

Sortida: es mostra l'import de la venda

En OCL

context: `Sistema::fiVenda(venda:Venda):Import`

body: `venda.import`

22

Exemple: operació *pagament*

Name: pagament(venda: Venda, importPag: Import): Import
Precondicions: importPag \geq venda.import
Postcondicions:
Sortida: es mostra el canvi a retornar = importPag - venda.import

En OCL

context: Sistema::pagament(venda:Venda, importPag:Import):Import
pre: importPag \geq venda.import
body: importPag-venda.import

23

Redundància - Definició

Una especificació és redundant si un mateix aspecte del sistema està definit/descriu diverses vegades.

La redundància dificulta la modificabilitat de l'especificació perquè si varia aquell aspecte cal modificar tots els models que hi fan referència.



L'especificació no ha de ser redundant !!!

Redundàncies possibles:

- Entre una precondició i l'esquema conceptual de les dades
- Entre una precondició i els diagrames de seqüència del sistema
- Entre una precondició i el diagrama d'estats (redundància que no veurem a IES)

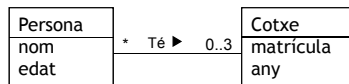
24

Redundància: Pre amb l'Esquema Conceptual de les Dades

L'esquema conceptual de les dades ja garanteix que les condicions imposades per les restriccions d'integritat (*estructurals, gràfiques i textuais*) mai es violaran



La precondició d'una operació no ha d'incloure les comprovacions necessàries per garantir que l'execució del cas d'ús no viola les restriccions d'integritat definides a l'esquema conceptual de les dades



Restriccions textuais:

1. Claus externes: (Persona,nom); (Cotxe,matricula)
2. Les persones menors de 18 anys no poden tenir cotxe

```

context: Sistema::comprarCotxe (nom, mat)
pre:
  Persona.allInstances()->exists(p|p.nom=nom
    and p.edat>18)
  Cotxe.allInstances()->exists(c|c.matricula=mat)
post:
  Persona.allInstances()->select(p|p.nom=nom).
    cotxe.matricula->includes(mat)
    
```

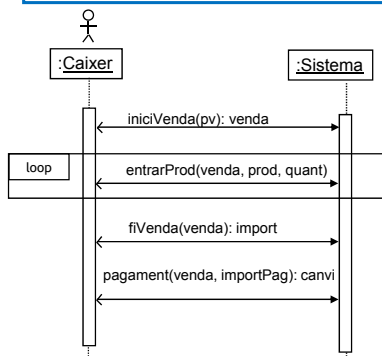
25

Redundància: Pre amb el Diagrama de Seqüència

Els diagrames de seqüència ja defineixen un ordre d'invocació de les operacions



Les operacions no han d'incorporar informació per garantir que aquest ordre es compleixi



Operació: Pagament (venda, importPag): canvi

Precondicions:

- ~~existeix venda "venda"~~
- ~~la venda "venda" ha finalitzat~~

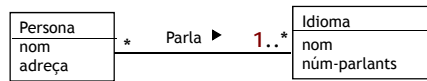
Postcondicions:

...

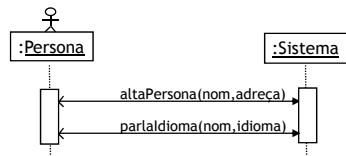
Sortida:

26

Exemple de diagrama de seqüència incorrecte

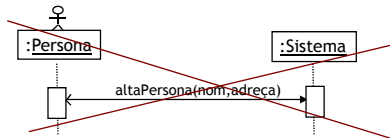


Cas d'ús: Afegir-Persona-1



Operació: `altaPersona(nom,adreça)`
 Precondicions:
 Postcondicions:
 - s'ha creat d'un objecte `Persona`
 amb el nom i l'adreça especificats
 Sortida:

Cas d'ús: Afegir-Persona-2

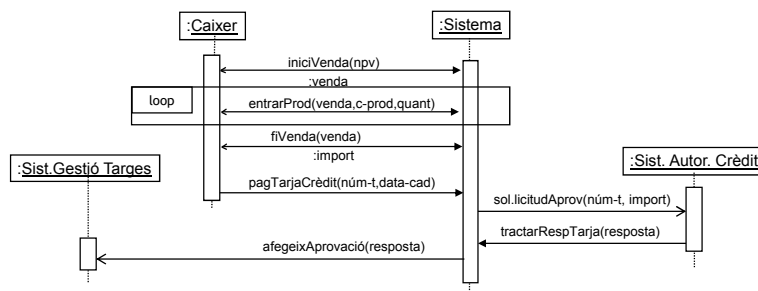


Operació: `parlaldioma(nom,nomId)`
 Precondicions:
 - l'idioma identificat per `nomId` existeix
 Postcondicions:
 - creació d'una associació `Parla` entre
 la persona amb `nom=nom` i l'idioma
`nom=nomId`
 Sortida:

27

Interacció amb actors software (serveis) externs

- De vegades un actor d'un cas d'ús pot ser un sistema software (servei) extern al sistema que estem desenvolupant.
- Exemple: cas d'ús comprar productes amb tarja de crèdit



La postcondició de l'operació `pagTarjaCrèdit` inclou la invocació del servei demanat al *Sistema d'Autorització de Crèdit*, mitjançant la frase:

‘El sistema invoca l'execució de l'operació `sol.licitudAprov` al sistema d'autorització de crèdit i li passa `núm-t` i `import` com a paràmetres

28

Redundància - Interpretació dels contractes

Com s'han d'interpretar els contractes en relació a l'esquema conceptual?

Precondicions:

què ha de contenir l'Esquema Conceptual per intentar executar una operació

Restriccions d'integritat:

s'han de satisfer després de l'execució de totes les operacions d'un cas d'ús



Suposem que es rebutgen totes les operacions d'un cas d'ús si la seva execució viola (globalment) alguna restricció d'integritat de l'esquema conceptual de les dades

29

Especificació en OCL de la sortida d'una operació

La sortida d'una operació acostuma a ser informació composta. O sigui, no elemental.

resumVendes(num-pv: Integer) retorna un llistat de vendes amb la informació:
Per cada Producte venut al PuntDeVenda num-pv mostrar el seu codi i el seu preu

```
resumVendes (num-pv:Integer): ???
```

Definició d'informació composta en OCL:

Tuples: composició de diversos elements, possiblement de tipus diferents

Definició del tipus: `TupleType(camp1:Tipus1,...,campN:TipusN)`

Assignació de valors: `Tuple{camp1=valor1,...,campN=valorN}`

Col·leccions: Set, Sequence i Bag.

```
resumVendes(num-pv:Integer):  
Set(TupleType(codiprod:Integer, preu:Import))
```

30

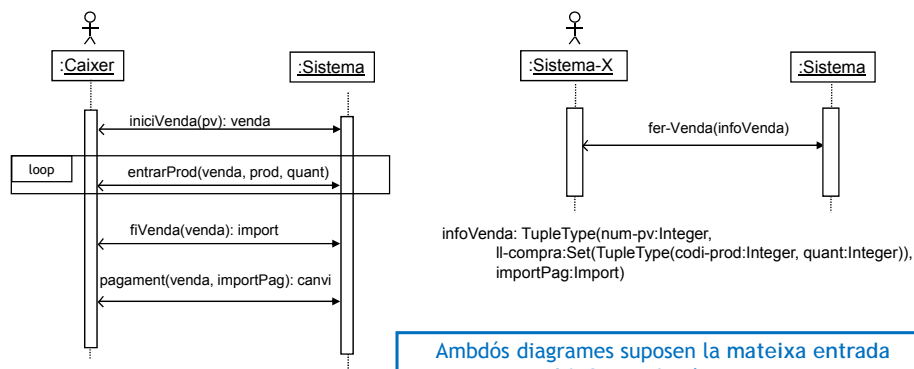
Exemple d'operació de consulta en OCL

```
context Sistema::resumVendes(npv:Integer):  
    Set(TupleType(cprod:Integer, preu: Import))  
  
pre: PuntDeVenda.allInstances()->exists(p|p.num-pv=npv)  
  
body:  
    let prods:Set(Producte) =  
        Producte.allInstances()->  
        select(p|p.liniaDeVenda.venda.puntDeVenda.num-pv->includes(npv))  
    in  
    prods->collect(p | Tuple{cprod = p.codi, preu = p.preu})
```

31

Diagrama de seqüència: quants esdeveniments?

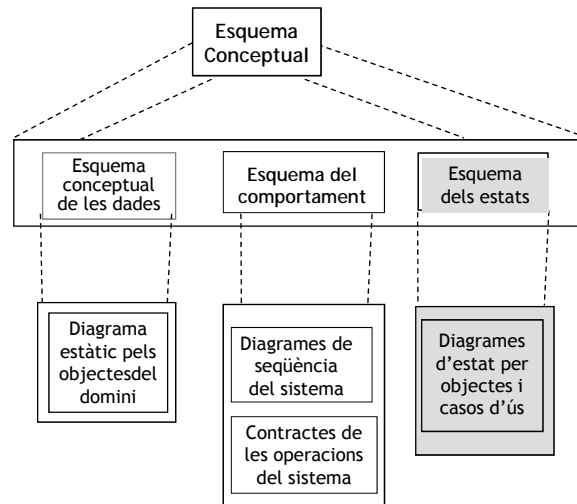
El nombre d'esdeveniments d'un diagrama de seqüència depèn de com es produeix la interacció entre els actors i el sistema software.



Ambdós diagrames suposen la mateixa entrada d'informació al sistema
Els dos diagrames de seqüència poden ser correctes, segons les circumstàncies

32

Esquema conceptual (especificació)



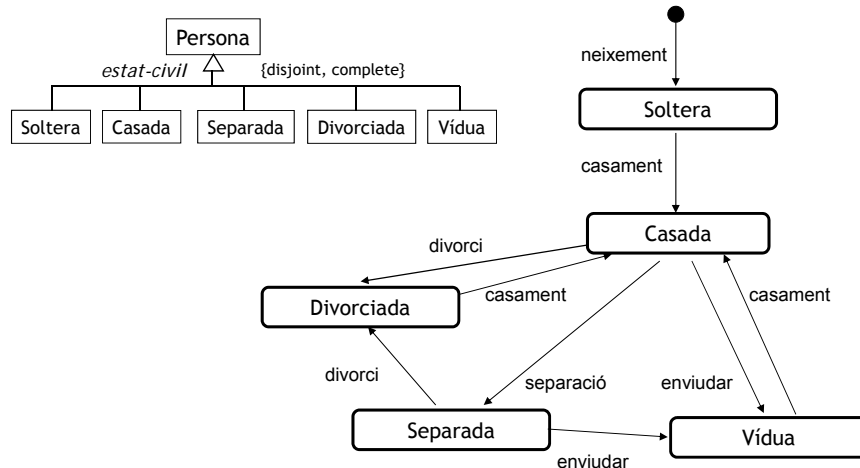
33

Esquema dels Estats

- Objectius:
 - crear diagrames d'estat per objectes i casos d'ús
- Estats, transicions i esdeveniments:
 - Estat:
 - condició d'un objecte o d'un cas d'ús en un moment del temps
 - Transició:
 - canvi d'estat com a conseqüència d'un esdeveniment
 - Esdeveniment:
 - tot allò que requereix una resposta del sistema software

34

Canvi d'estat civil d'una persona



35

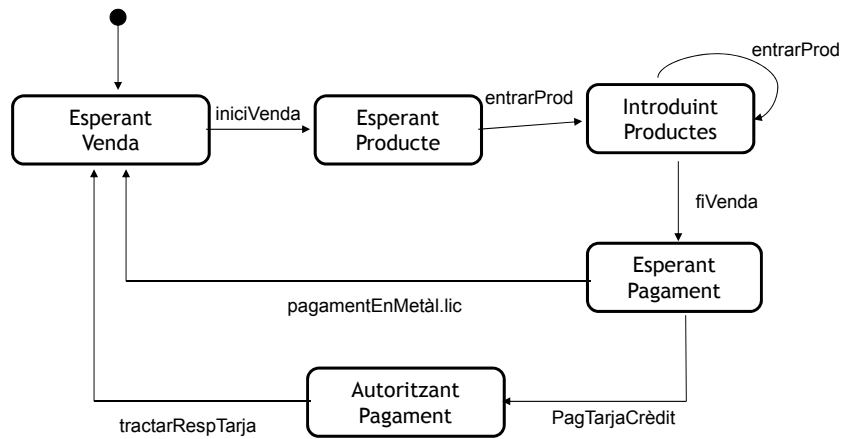
Ús dels diagrames d'estat

- Un diagrama d'estats es pot especificar per a una:
 - Classe d'objectes:
 - per descriure perquè els objectes canvien de subclasse
 - les subclasses d'un diagrama d'estats no tenen perquè aparèixer explícitament a l'esquema conceptual
 - per descriure classes d'objectes amb important "comportament dinàmic"
 - Cas d'ús:
 - per descriure la seqüència legal en la que els esdeveniments es poden produir al món real

p. ex. en una compra de producte no es pot fer el pagament fins que no s'hagi tancat la venda.

36

Diagrama d'estats del cas d'us "comprar productes"



37

Bibliografia

- Larman, C. *"Applying UML and Patterns. An Introduction to Object-oriented Analysis and Design"*, Prentice Hall, 2005, (3^a edició)
- Rumbaugh, J.; Jacobson, I.; Booch, G. *"The Unified Modeling Language Reference Manual"*, 2^a edició, Addison-Wesley, 2004
- UML 2.3 Superstructure Specification
<http://www.omg.org/spec/UML/2.3/Superstructure/PDF>
OMG, 2010

38