ARQUITECTURA DEL SOFTWARE

Unit 2.2: Object Oriented Architecture

Enunciats Exercicis

Exercici 1

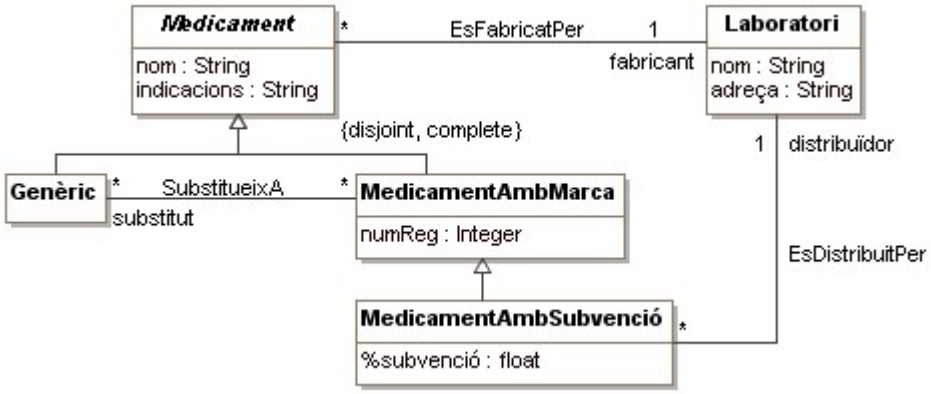
Un dels principis de disseny que han de complir les arquitectures orientades a objecte és el baix acoblament. Fes una taula on es descrigui el tipus d'acoblament (mireu *Ingeniería del software. Un enfoque práctico.* R.G. Pressman. McGraw Hill, 2010), en quin/s diagrames es pot detectar el tipus d'acoblament i una possible manera d'evitar-ho.

Exercici 2

Considera el diagrama de classes de la transparència 13 del tema 2.2. Explica, de forma textual (no cal fer el contracte), com es crea una venda (*Sale*).

Exercici 3

Una agrupació de laboratoris farmacèutics ens ha demanat que dissenyem una part del seu sistema informàtic per gestionar els medicaments que serveix a les farmàcies. Els laboratoris d'aquesta agrupació s'encarreguen de fabricar i distribuir medicaments. Els medicaments que fabriquen poden ser genèrics o amb marca. Cada medicament genèric pot ser substituït per medicaments amb marca i al contrari. Alguns medicaments amb marca són subvencionats per la seguretat social. Aquests últims són distribuïts per laboratoris de l'agrupació (que poden ser diferents del laboratori de fabricació). A continuació disposeu de l'esquema conceptual i dels contractes de les operacions a dissenyar.



R.Integritat Textuals:
- Claus classes no associatives: (Medicament, nom); (Laboratori, nom)

context CapaDomini::laboratoris(nom:String):Set(String)
exc *medicament-no-existeix*: el medicament amb el nom *nom* no existeix.
post result= si *nom* és un medicament amb subvenció aleshores es retorna el nom del laboratori que el distribueix. Si és un medicament amb marca però no subvencionat, es retorna el nom dels laboratoris que fabriquen els medicaments genèrics que substitueixen a *nom*. Si és un genèric el seu laboratori de fabricació.

context CapaDomini :: esborrarMedicamentAmbMarca(nom:String)
exc *medicamentAmbMarca-no-existeix*: el medicament amb marca amb el nom *nom* no existeix.

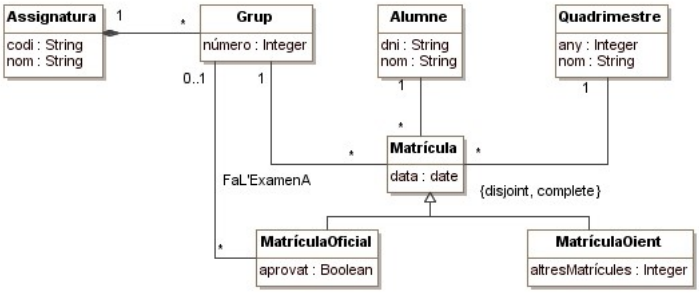
post S'eliminen les instàncies de l'associació *SubstitueixA* entre el medicament i els genèrics substituts. Si el medicament és amb subvenció, a més, s'elimina la instància de l'associació entre el medicament *nom* i el laboratori que el distribueix.
post s'elimina la instància de l'associació entre el medicament *nom* i el laboratori que el fabrica.
post s'elimina el medicament *nom*.

Es demana:

a) Diagrama de seqüència de l'operació *laboratoris* i *esborrarMedicamentAmbMarca*. S'ha de justificar l'aplicació dels principis de disseny estudiats.
b) Diagrama de classes de la capa de domini.

Exercici 4

Un centre docent ens ha encarregat el disseny d'alguns casos d'ús per a l'ajuda al seu procés de matriculació. El centre oferta cada quadrimestre un conjunt d'assignatures. Cada assignatura disposa d'un conjunt de grups. Un alumne pot tenir diverses matrícules en un quadrimestre per diferents assignatures. Les matrícules poden ser oficials o d'oients (sense dret a examen). Les matrícules oficials enregistren si l'alumne en acabar el curs ha aprovat l'assignatura i en quin grup ha fet l'examen. Les matrícules d'oient enregistren el nombre de matrícules totals que té l'alumne aquell mateix quadrimestre. A continuació disposeu de l'esquema conceptual i dels contractes de les operacions a dissenyar:



R.Integritat Textuals:
- Claus classes no associatives: (Assignatura, codi); (Alumne, dni); (Quadrimestre, nom)
- No poden haver-hi dues matrícules pel mateix grup d'assignatura, alumne i quadrimestre
- Una assignatura no pot tenir dos grups amb el mateix número
- L'assignatura del grup on té dret a fer l'examen un alumne amb matrícula oficial ha de ser la mateixa que l'assignatura del grup on s'ha matriculat
- Els alumnes amb matrícula oficial que han aprovat l'assignatura han fet l'examen en algun grup de l'assignatura
- Un alumne no es pot matricular a dos grups de la mateixa assignatura.
- ...altres restriccions no rellevants pel problema

context CapaDomini::repetidors(codAss: String, nomQuad: String): Set(String)
exc *assignatura-no-existeix*: l'assignatura *codAss* no existeix
exc *quadrimestre-no-existeix*: el quadrimestre *nomQuad* no existeix

post resultat= nom dels alumnes que han matriculat l'assignatura *codAss* al quadrimestre *nomQuad* i que o són oients o no han aprovat l'assignatura o havent-la aprovat no van fer l'examen al mateix grup on es van matricular.

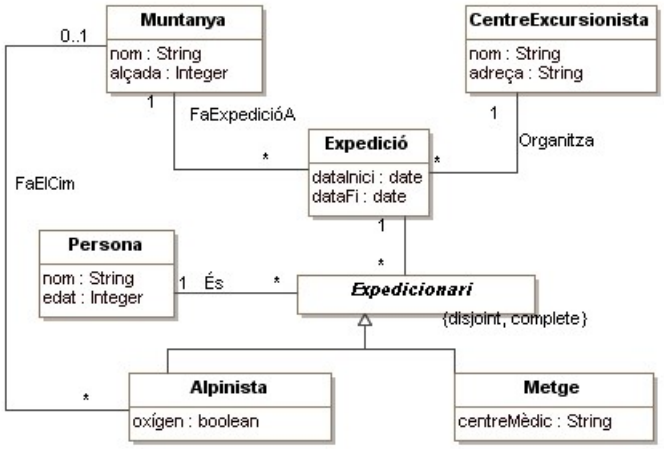
context CapaDomini::baixa(codAss: String, numG: Integer, dni: String, nomQuad: String)
exc *matrícula-no-existeix*: la matrícula identificada per *codAss, numG, dni i nomQuad* no existeix
post S'elimina la matricula i les associacions amb el grup, alumne i quadrimestre
post Si la matricula és oficial, s'elimina l'associació amb el grup on s'ha fet l'examen (si és necessari)

Es demana:

a) Diagrama de seqüència de les operacions *repetidors* i *baixa*. S'ha de justificar l'aplicació dels principis de disseny estudiats.
b) Diagrama de classes de la capa de domini.

Exercici 5

Una agrupació de centres excursionistes ens ha encarregat el disseny d'uns casos d'ús per enregistrar la informació de les expedicions que organitzen. Els centres excursionistes organitzen expedicions a muntanyes en un període determinat. A les expedicions participen expedicionaris (alpinistes o metges). Pels alpinistes s'enregistra si han fet el cim a una muntanya determinada. A continuació disposeu del diagrama de classes de la capa de domini i dels contractes de les operacions a dissenyar:



R.I. Textual:
- Claus classes: (Centre-exc., nom); (Muntanya, nom); (Persona, nom)
- La data fi d'una expedició ha de ser posterior a la seva data d'inici.
- No poden haver-hi dues expedicions del mateix centre exc. i per a la mateixa muntanya en períodes solapats.
- No poden haver-hi dos expedicionaris per la mateixa persona i expedició.
- Un alpinista pot fer el cim d'una muntanya si ha participat en una expedició per aquella muntanya
- ... altres restriccions no rellevants pel problema

context CapaDomini::alpinistesTop(nomM: String): Set(String)
exc *muntanya-no-existeix*: la muntanya *nomM* no existeix
post retorna el nom dels alpinistes que han intentat fer el cim a la muntanya *self* com alpinistes una única vegada i ho han aconseguit.

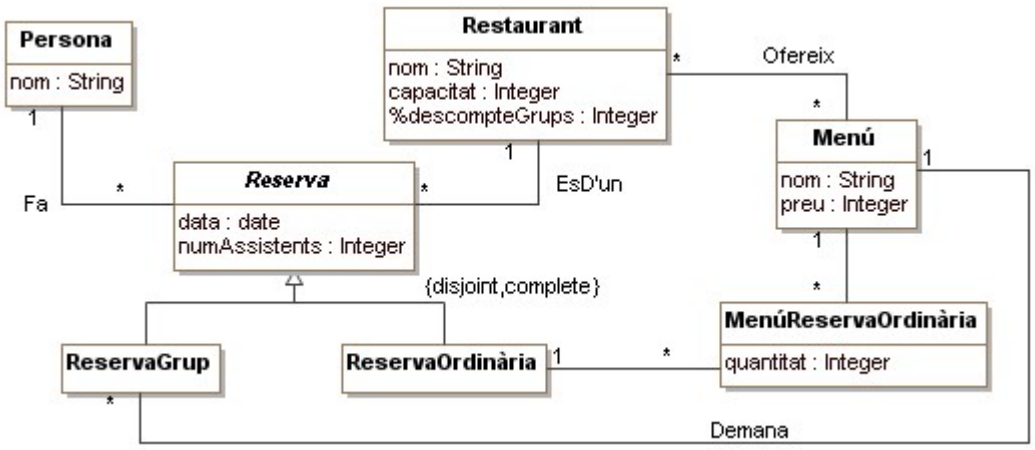
context CapaDomini::baixa(nomP:String)
exc *persona-no-existeix*: la persona *nomP* no existeix
exc *expedició-en-curs*: la persona està en una expedició (és a dir, Data-ini < DataActual < Data-fi).
exc *no-té-expedicions*: la persona no ha fet cap expedició.
post es dóna de baixa la persona i també es dóna de baixa totes les seves participacions com a expedicionari en expedicions.
post s'eliminen les instàncies corresponents de l'associació fa-el-cim, si la persona havia fet el cim com alpinista en expedicions.

Es demana:

a) Diagrama de seqüència de les operacions *alpinistesTop* i *baixa*. Podeu suposar que la *DataActual* es pot obtenir del sistema invocant l'operació *getDataActual():Date*. S'ha de justificar l'aplicació dels principis de disseny estudiats.
b) Diagrama de classes de la capa de domini.

Exercici 6

Una cadena de restaurants que ofereixen menús per sopars ens ha demanat que li dissenyem una part d'un sistema software per gestionar les seves reserves. Els clients de la cadena fan reserves per un restaurant concret en una data determinada. Les reserves poden ser de dos tipus: les reserves de grup són reserves per a un conjunt de persones que demanen el mateix menú i ho fan en el moment de fer la reserva, i les reserves ordinàries són reserves per a una o més persones que demanen el menú que vulguin en el moment de sopar. Els restaurants ofereixen un descompte (%descompteGrups) en el preu del menú per a les reserves de grup. A continuació disposeu de l'esquema conceptual i dels contractes de les operacions a dissenyar:



R.I. Textuals:
- Claus: (Persona, nom); (Restaurant, nom); (Menú, nom); (Reserva, nomPersona+data)
- No poden haver-hi dos menús de reserva ordinària pel mateix menú i reserva ordinària.
- Els menús de les reserves han de ser menús oferts pel restaurant on s'ha fet la reserva.
- La capacitat d'un restaurant ha de ser més gran que el sumatori dels assistents de les reserves d'aquell restaurant per una data determinada.
- Tots els atributs de tipus integer han de ser positius
- El numAssistents d'una reserva de grup ha de ser més gran que 5
- En una reserva ordinària el numAssistents ha de coincidir amb la quantitat de menús demanats
- Altres restriccions no rellevants pel problema

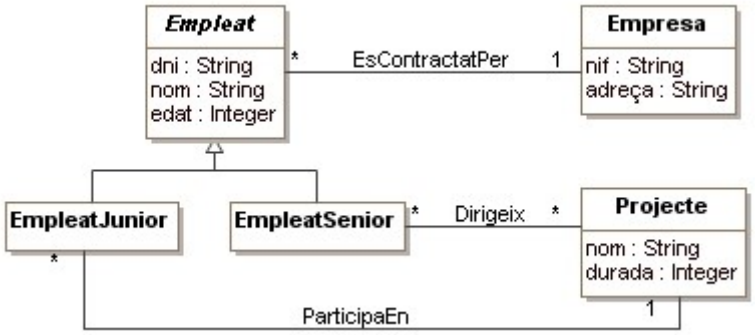
context CapaDomini::preuReserva(nomP: String, data:date): Float
pre *té-reserva*: la persona té una reserva a la data indicada al paràmetre.
post result= preu de la reserva associada a la persona *self* en la data indicada al paràmetre. El preu d'una reserva es calcula de la següent forma:
1) si la reserva és de grup aleshores preu= numAssistents*preu menú demanat - %descompteGrups del restaurant/100*(numAssistents*preu menú demanat)
2) si la reserva és ordinària aleshores preu= ∑ quantitat*preu menú per a cada menú demanat.

context CapaDomini::novaReservaGrup(nomP: String, nomR: String, dr: date, nAss: Integer, nomMenú: String)
pre *persona-existeix*: la persona *nomP* existeix.
pre *restaurant-existeix*: el restaurant *nomR* existeix.
pre *numAssistents-ok*: el *nAss* és més gran que 5.
pre *restaurant-amb-capacitat*: el restaurant *r* té capacitat per acollir la reserva.
exc *ja-existeix-reserva*: la persona ja ha fet una reserva per la data *dr*.
exc *menú-no servit*: el menú *nomMenú* no es ofert pel restaurant *nomR*.
post es crea una reserva de grup i es formen totes les associacions amb la persona, el restaurant i el menú.

Es demana:
a) Diagrama de seqüència de l'operació *preuReserva* i *novaReservaGrup*.
b) Diagrama de classes de la capa de domini.

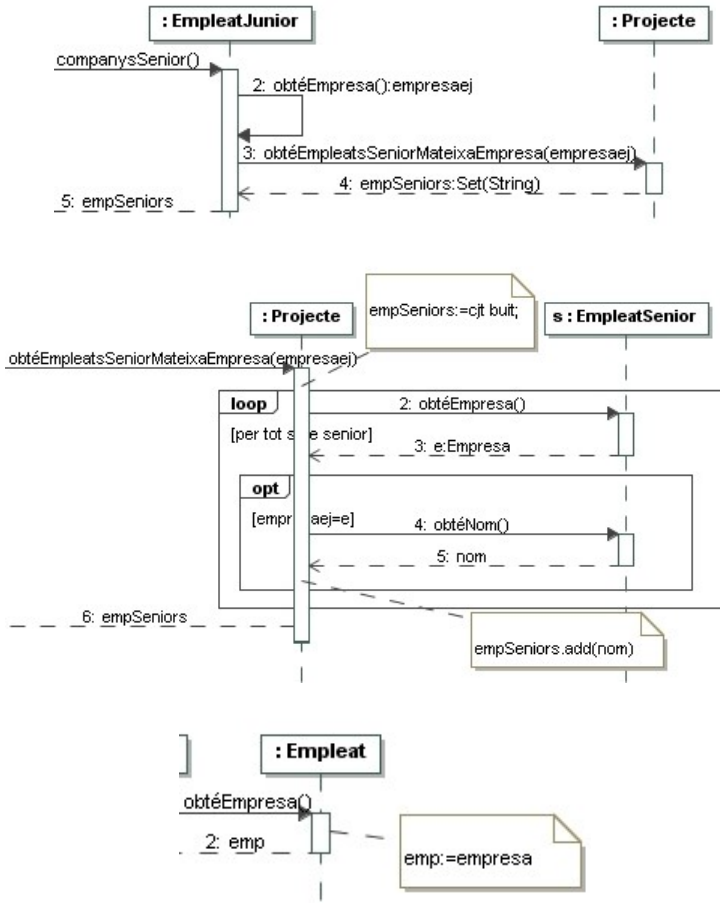
Exercici 7

(Competència transversal) Donat l'esquema conceptual i els diagrames de seqüència següents:



R.Integritat Textuals:
- Claus classes no associatives: (Empresa, nif); (Empleat, dni); (Projecte, nom)

context Junior::companysSenior(): Set(String)
post resultat= nom dels empleats seniors que dirigeixen el projecte on *self* participa i que estan contractats per la mateixa empresa.

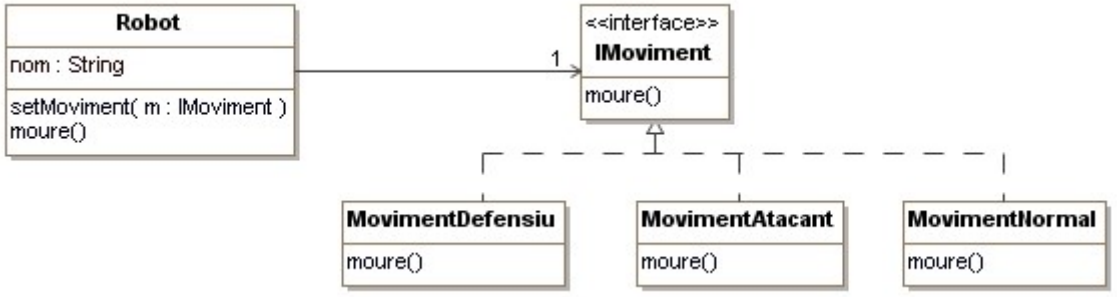


Es demana:

a) Raona i justifica si el diagrama de seqüència anterior aplica el principi de disseny de *baix acoblament*.
b) Proposa una solució alternativa a l'anterior que millori l'acoblament.

Exercici 8

Un laboratori de disseny de robots ens ha demanat que li dissenyem una part d'un sistema software per simular els diferents moviments que fan els robots que dissenyen. Els robots tenen un nom i realitzen moviments en funció d'una estratègia de moviment (que pot ser defensiva, atacant o normal). A continuació disposeu del diagrama de classes del paquet *domain model* de la capa de domini i dels contractes d'un conjunt d'operacions de la capa de domini.



R.I. Textuals:
- Claus: (Robot, nom)

context CapaDomini::crearRobotAmbMovimentsDefensius (nom:String)
exc *robot-existeix*: el robot amb nom *nom* existeix.
post *crea-robot***:** es crea el robot amb nom *nom* amb moviments defensius.

context CapaDomini::canviarAMovimentAtacant (nom:String)
exc *robot-no-existeix*: el robot amb nom *nom* no existeix.
post *modifica-moviment***:** es crea un el moviment atacant i s'assigna al robot.

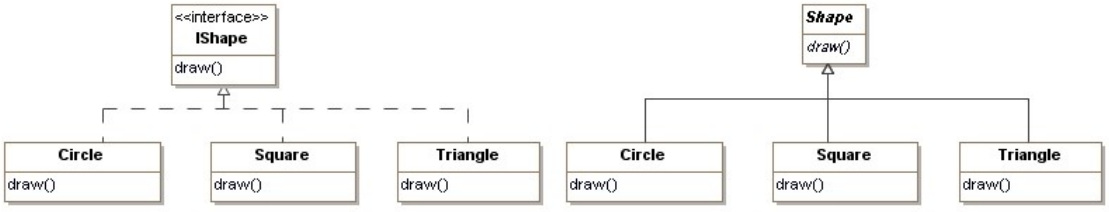
context CapaDomini::moureRobot(nom:String)
exc *robot-no-existeix*: el robot amb nom *nom* no existeix.
post es mou el robot segons el moviment que tingui assignat.

Es demana:

a) Diagrama de seqüència de les operacions anteriors.

Exercici 9

(Competència transversal) Ens han demanat que dissenyem un editor gràfic que ha de tenir funcionalitats per dibuixar figures senzilles. A continuació disposeu de dues alternatives d'un fragment del diagrama de classes del paquet *domain model* de la capa de domini.



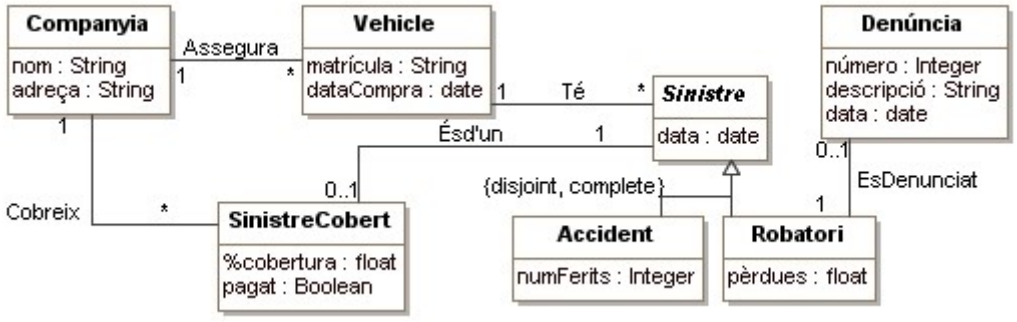
Es demana:

a) Raona i justifica quina de les alternatives et sembla més adient en els següents supòsits:
   a. Tal i com es mostra a la figura, la implementació de l'operació *draw* és diferent per a cada figura.
   b. L'operació *draw* es modifica i passa a tenir una part igual per a totes les figures i una altra part diferent per a cada una d'elles.
   c. Està previst en un futur afegir l'atribut privat *center* (que indica el punt central de la figura) a *Shape*.
   d. Està previst en un futur afegir noves operacions públiques a *Circle, Square i Triangle* que heretin d'una classe A.

Exercici 10

Un consorci de companyies d'assegurances ens ha demanat que li dissenyem una part d'un sistema software per gestionar els sinistres dels vehicles que tenen assegurats. Els vehicles s'identifiquen per matrícula i es guarda la seva data de compra i la companyia a la que estan assegurats. Un vehicle pot tenir un sinistre en una determinada data. Un sinistre pot estar cobert per una companyia asseguradora. Si ho està, se'n coneix el

percentatge de cobertura i si ha estat pagat o no. A més, de tots els sinistres possibles, interessa guardar informació específica dels accidents (es vol conèixer el nombre de ferits, si n'hi ha) i dels robatoris (es vol saber l'import de les pèrdues i si han estat denunciats o no). Les denúncies s'identifiquen per un número i se'n coneix també la data en què es van fer efectives i la seva descripció. A continuació disposeu de l'especificació feta per a aquest sistema.



R.I. Textuals:
- Claus: (Companyia, nom); (Vehicle, matrícula); (Sinistre, matrícula, data); (Denúncia, número); (SinistreCobert, nom, matrícula, data)
- La companyia asseguradora que cobreix el sinistre ha de ser la companyia on està assegurat el vehicle.
- Els sinistres coberts que corresponen a robatoris només poden ser pagats si tenen la denúncia corresponent.
- La data de la denúncia ha de ser posterior a la data del sinistre.
- Altres restriccions no rellevants pel problema

context CapaDomini::llistarVehicles(nomComp:String, data:Date): SetVehicles on SetVehicles={matrícula}
pre *companyia-existeix*: la companyia *nomComp* existeix.
post result= conjunt de matrícules dels vehicles assegurats per la companyia *nomComp* que tenen algun sinistre en data posterior a *data* amb una cobertura superior al 50% i que són de tipus robatori denunciat o bé de tipus accident sense ferits.

context CapaDomini::canviCompanyiaAsseguradora(matr:String, novaComp:Companyia)
exc *sinistres-no-pagats*: el vehicle té sinistres coberts per la companyia actual que no han estat pagats.
post es dóna de baixa la instància de l'associació entre la companyia actual i el vehicle *matr*.
post s'eliminen tots els sinistres que ha tingut el vehicle *matr*, així com les denúncies, si en té, pel cas dels sinistres amb robatori.
post s'eliminen totes les instàncies de sinistre cobert que fan referència a la companyia actual i al sinistre del vehicle *matr*.
post es dóna d'alta la instància de l'associació entre la companyia *novaComp* i el vehicle *matr*.

Es demana:

a) Diagrama de seqüència de l'operació *llistarVehicles* i *canviCompanyiaAsseguradora*. S'ha de justificar l'aplicació dels principis de disseny estudiats.
b) Diagrama de classes de la capa de domini.