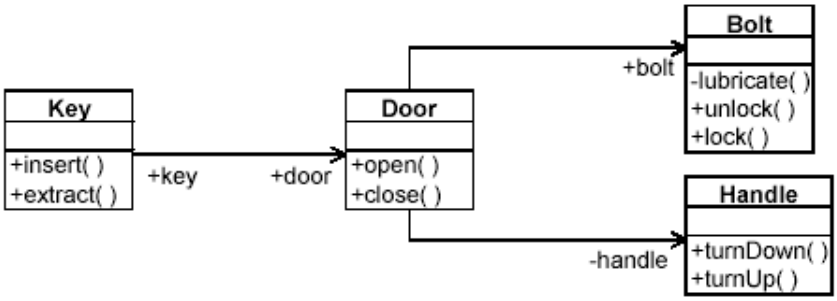
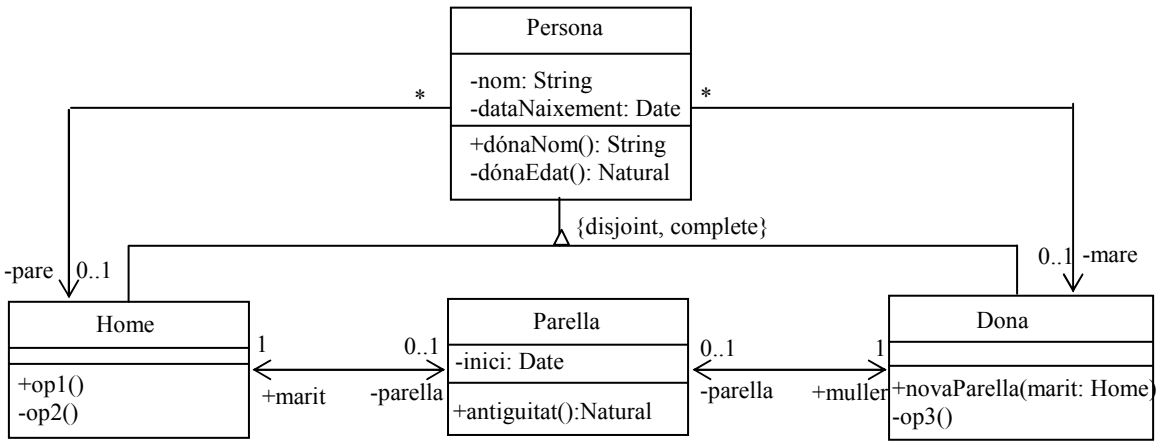
Tema 3 – Patró arquitectònic: Orientació a Objectes

1. Considereu el fragment de disseny indicat a la figura, i digueu:
   (a) Quines operacions poden ser invocades des d'un objecte de la classe *Door*.
   (b) Quines operacions poden ser invocades des d'un objecte de la classe *Bolt*.
   (c) Quines operacions poden ser invocades des d'un objecte de la classe *Handle*.
   (d) Quines operacions poden ser invocades des d'un objecte de la classe *Key*.



En respondre a la pregunta, heu de considerar exclusivament la navegabilitat de les associacions.

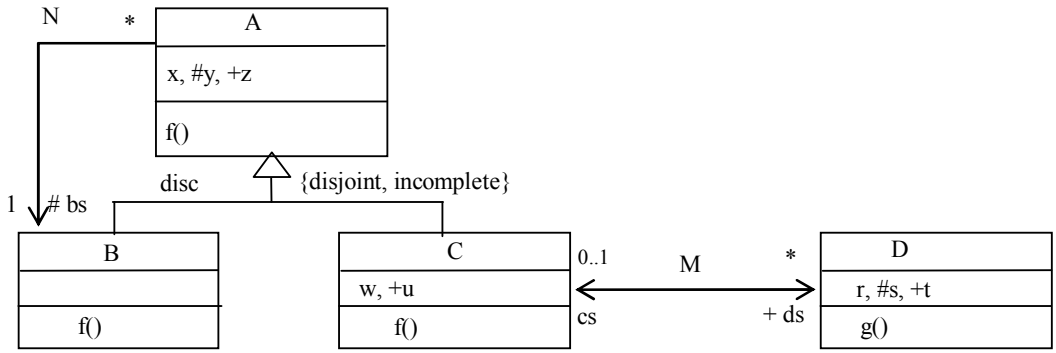
2. Considereu el fragment (parcial) de disseny indicat a la figura, i digueu:
   (a) Feu una taula que mostri, per cada classe, quins atributs, rols i operacions són visibles.
   (b) Digueu les diverses possibilitats que té un objecte de la classe *Home* de cridar les operacions que acabeu d'establir que són visibles des de la classe *Home*.
   (c) Ídem per a la classe *Parella*.



En respondre a la pregunta, heu de considerar exclusivament la navegabilitat de les associacions.

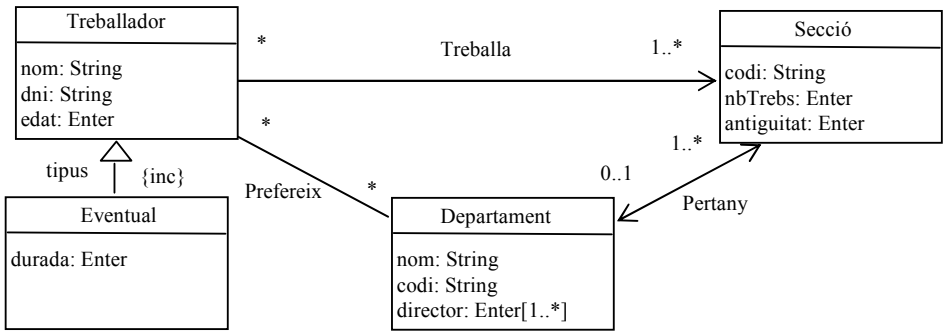
3. En base al diagrama de classes de disseny en UML de la figura, responeu a les preguntes següents:
   (a) Digueu quins atributs són accessibles desde cada una de les classes.
   (b) Digueu si la afirmació següent és certa i justifiqueu la resposta: "Una operació de *B* pot invocar a una operació polimòrfica d'*A* encara que no hi hagi navegabilitat de *B* cap a *A*". Quines invocacions de *f()* pot efectuar un objecte *b* de classe *B*? Ídem per a objectes *c* i *d* de classes *C* i *D*.

(c) Digueu quines de les quatre classes poden ser abstractes. Quins canvis s'haurien de fer en el model per tenir-ne més?

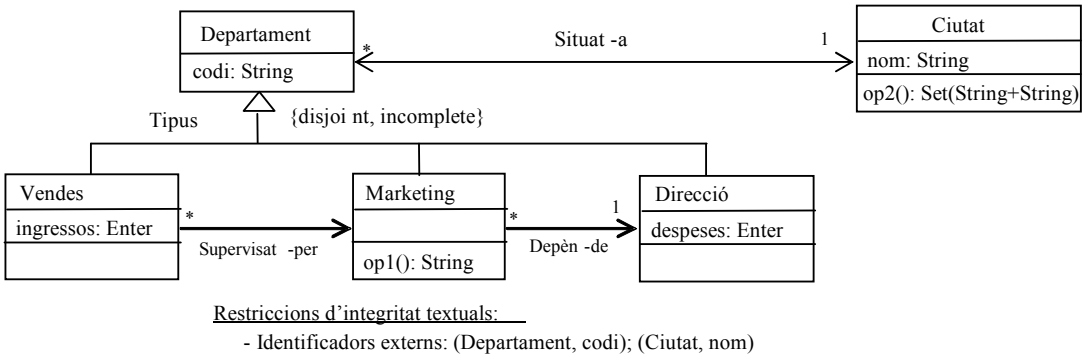


En respondre a les preguntes, heu de considerar exclusivament la navegabilitat de les associacions.

4. Declareu TOTES les operacions *getter*, *setter* i creadores en el diagrama de classes de disseny següent:



5. En base al diagrama de classes de disseny en UML de la figura i sense possibilitat de modificar-lo, respongueu a les preguntes següents.



Restriccions d'integritat textuals:

- Identificadors externs: (Departament, codi); (Ciutat, nom)

(a) Digueu i justifiqueu si l'afirmació "la classe *Departament* no pot tenir cap operació polimòrfica perquè és una classe concreta i la seva jerarquia d'especialització és incomplete" és certa o falsa.

(b) L'operació *op1()* de la classe *Marketing* retorna el nom de la ciutat on està situat el departament de direcció del qual depèn. És possible dissenyar aquesta operació amb aquest diagrama de classes? En cas afirmatiu, feu el diagrama de seqüència.

En cas negatiu, justifiqueu perquè no. Quin és l'acoblament introduït per la vostra solució?

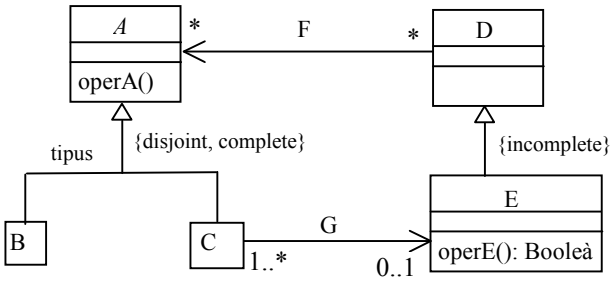
(c) L'operació *op2()* de la classe *Ciutat* retorna el codi i nom de la ciutat dels departaments de *Marketing* que supervisen departaments de *Vendes* que estan situats a la ciutat. És possible dissenyar aquesta operació amb aquest diagrama de classes? En cas afirmatiu, feu el diagrama de seqüència. En cas negatiu, digueu perquè no. Quin és l'acoblament introduït per la vostra solució?

6. Tenint en compte el diagrama de classes de la figura, digueu si són certes o falses les afirmacions següents. Justifiqueu breument la resposta.

(a) Per tal de garantir les restriccions de multiplicitat de l'associació *G*, al crear un objecte de la classe *E* també s'ha de crear un objecte de la classe *C*.

(b) Els objectes de la classe *C* poden crear objectes de la jerarquia *DE* únicament si aquests es creen com objectes de la classe *E*.

(c) Si l'operació *operA()* de la classe *A* crea una instància de la classe *D* llavors la navegabilitat de l'associació *F* passa a ser bidireccional.



7. Hi ha un principi de disseny anomenat principi d'Obert-Tancat (OCP). Busqueu-ne informació i feu-ne un resum. Enumereu dos constructors OO que facilitin l'assoliment d'aquest principi i poseu-ne un exemple.

8. Un dissenyador novell ens proposa una part del diagrama de classes d'un sistema per a enregistrar els vehicles aparcats en els aparcaments d'una empresa d'aparcaments. Cada cop que el vehicle aparca acumula un punt, els quals serveixen per a obtenir futurs descomptes i regals. Per a aparcar, cal recollir prèviament un tiquet a l'entrada que servirà per pagar l'import al retirar el vehicle (aspecte no mostrat en el disseny per simplificar el problema); les operacions *teTiquet()* i *haPagat()* permeten comprovar aquestes condicions.

Es demana:

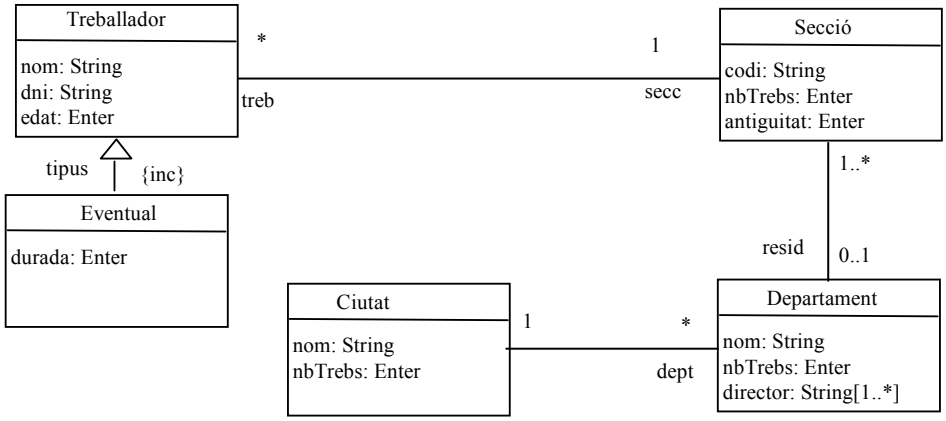
(a) Feu els contractes de les operacions *aparcarVehicle()* i *retirarVehicle*(). Considereu que *nom* i *matr* són clau de les seves classes.

(b) En el cas d'operacions polimòrfiques, quina relació hi ha entre les condicions que han de complir les pre i postcondicions dels contractes d'aquestes operacions descrites a classe (Meyer) amb el Principi de Substitució de Liskov (LSP). Aquest principi el podeu trobar explicat en diferents setis de la web consultant via Google

(per exemple www.objectmentor.com/resources/articles/lsp.pdf) i en els llibres de la bibliografía (Martin, Larman, ...).

(c) Suposem que el dissenyador vol introduir una especialització de la classe *Aparcament*, distingint entre els que són de pagament (*Aparcament*) i els que són gratuïts (*AparcaGratis*). Aquests darrers no tenen gestió de tiquets ni donen punts. Modifiqueu la solució d'(a) per recollir aquesta situació.

9. Sigui el model següent ja normalitzat:



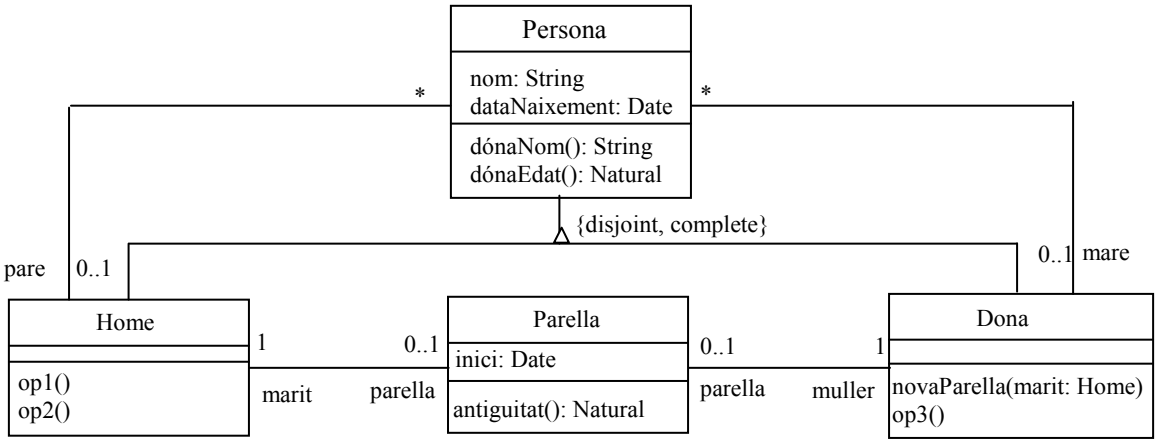
(a) Construïu els diagrames de seqüència de les operacions següents:

```
context Ciutat::nbDepts(): Enter
post result = nombre de departaments que resideixen a self

context Treballador::nomCiutat(): String
post result = nom de la ciutat on resideix el departament al que
              pertany la secció on treballa self (o null si
              la secció no pertany a cap departament)

context Treballador::canviSecció(novaSecc: Seccio)
pre novaSecc és diferent de la sección on self treballa actualment
pre tant la secció on treballava el treballador com la secció novaSecc,
    resideixen a un departament
post es decrementa nbTrebs de la secció on treballava el treballador
post es decrementa nbTrebs de la ciutat on resideix el
     departament al que pertany la secció on treballava la persona
post el treballador passa a estar associat amb la secció novaSecc
post s'incrementa nbTrebs de la ciutat on resideix el
     departament al que pertany la secció novaSecc
post es decrementa nbTrebs del departament al que pertany la secció on
     treballava la persona
post s'incrementa nbTrebs de la secció novaSecc
post s'incrementa nbTrebs del departament al que pertany la secció novaSecc

context Treballador::quiAnyPassaAnyEmpeny()
post self.edat++
post si self és de tipus eventual, llavors durada--

context Treballador::PromocionarAdirector()
pre self no és director ja del seu departament
post afegeix el nom de self com a director del departament corresponent
     si és que la secció on treballa self és d'algun departament

context Eventual::nomDept(): String
pre la secció del treballador eventual self resideix a un departament
post result = nom del departament de la secció de self
```

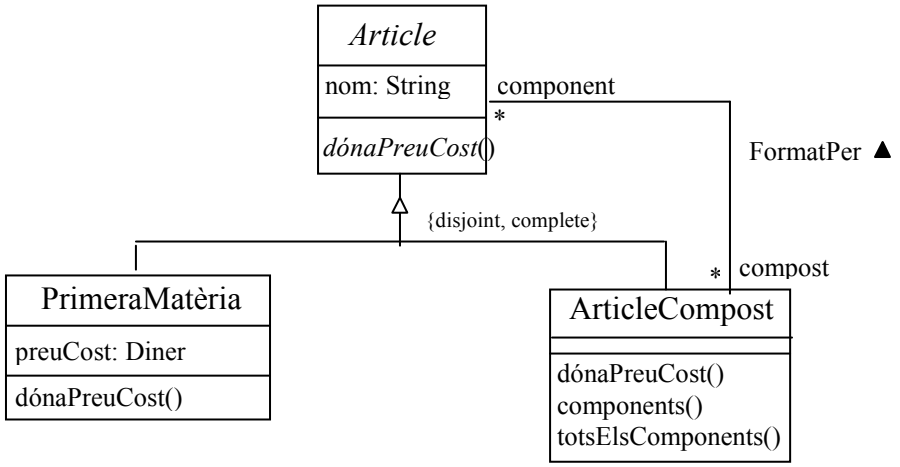
(b) Declareu totes les operacions que hagin aparegut en la solucio.

(c) Quin és l'acoblament resultant de la vostra solució? És possible reduir-lo? En cas afirmatiu, com i amb quines conseqüències?

10. Definiu el contracte de l'operació *Dona*::*novaParella*(*marit*: *Home*). La precondició és que ni la dona ni el marit no tinguin ja una parella. La postcondició ha d'establir que s'ha creat una parella nova, amb les associacions necessàries, i data d'inici *currentDate* (que ja està definida com la data del moment de l'execució de l'operació). És possible que la formalització d'aquesta operació requereixi la inclusió de noves operacions al fragment anterior. Si és així, indiqueu on anirien.



11. En relació al fragment del model de disseny següent:




```
context PrimeraMatèria::dónaPreuCost():Diner
      post: result = preuCost

context ArticleCompost::dónaPreuCost():Diner
      post: result = suma dels preuCost dels components

context ArticleCompost::components(): Set(Article)
      post: result = component

context ArticleCompost::totsElsComponents(): Set(Article)
      post: result = tots els components directes o indirectes
```

A més considereu que totes les operacions tenen les precondicions següents, procedents de l'especificació:

```
El nom d'article és clau
Un article compost no pot ser component de si mateix
```

(a) Indiqueu els diagrames de seqüència de les operacions *components*() i *totsElsComponents*(). Quina és la navegabilitat resultant de les associacions del diagrama?

(b) Definiu el contracte de l'operació:

```
ArticleCompost::afegeixComponent(nouComponent: Article)
```

(c) Definiu el contracte de l'operació:

```
ArticleCompost::treuComponent(componentExistent: Article)
```

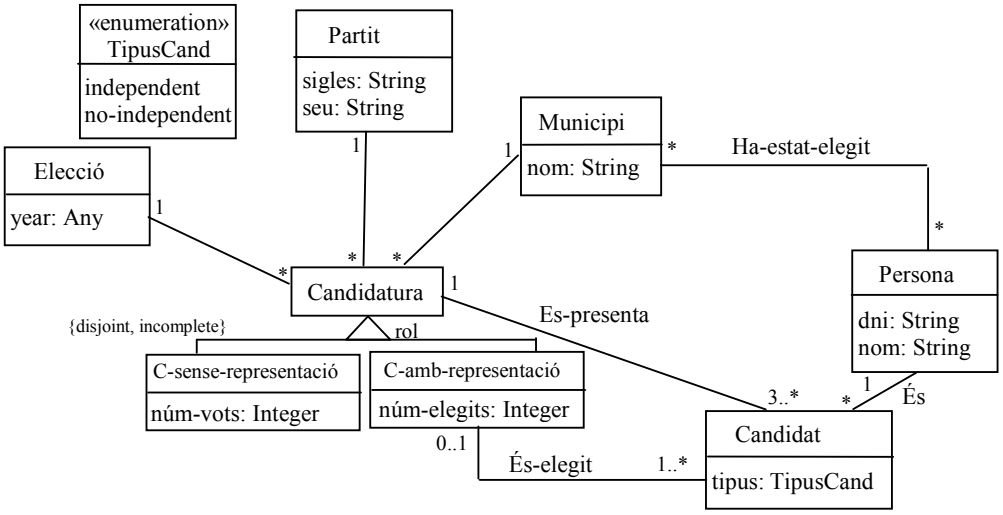
(d) Suposeu que afegim l'operació:

```
Article::enQuantsCompostosIntervens(): Natural
```

que retorna el nombre de compostos (pot ser zero) en què intervé l'article (directament o indirecta). Definiu el contracte d'aquesta operació.

En les respostes d'aquestes preguntes, podeu afegir noves operacions si s'escau, que heu d'especificar també.

12. En relació al fragment del model de disseny indicat a la figura següent:



Restriccions textuals:

- Un *Partit* s'identifica per les seves sigles. Un *Municipi* pel seu nom. Una *Elecció* pel seu any i una *Persona* pel seu dni.
- Un candidat només es pot elegir a la candidatura a la que es presenta.
- Una persona no es pot presentar a més d'una candidatura per elecció.

A més, en el model d'especificació es disposava de les dues regles d'informació derivada següents:

- núm-elegits=nombre de candidats elegits per una candidatura.
- Ha-estat-elegit: una persona ha estat elegida alguna vegada a un municipi.

(a) Definiu el contracte de l'operació de classe:

```
Candidatura::altaCandidatura(sigles-p: String, any-e: Any,
                  nom-m: String, ll-pers: Set(String)): Boolean
```

La precondició ha d'exigir que l'operació només es pot invocar si existeix un objecte *Elecció* identificat per *any-e*. L'operació ha de retornar fals si no existeix el partit identificat per *sigles-p* o no existeix el municipi identificat per *nom-m* o ja

existeix una candidatura d'aquell partit a aquell municipi en aquell any o no existeix la persona corresponent d'algun dni de *ll-pers*. En cas contrari l'operació ha de retornar cert. La postcondició ha d'assegurar que s'ha creat una candidatura associada amb els objectes elecció, partit i municipi i que s'han creat tantes instàncies de *Candidat* no independent com dnis hi hagi a *ll-pers* amb les corresponents associacions *Es-presenta* i *És*.

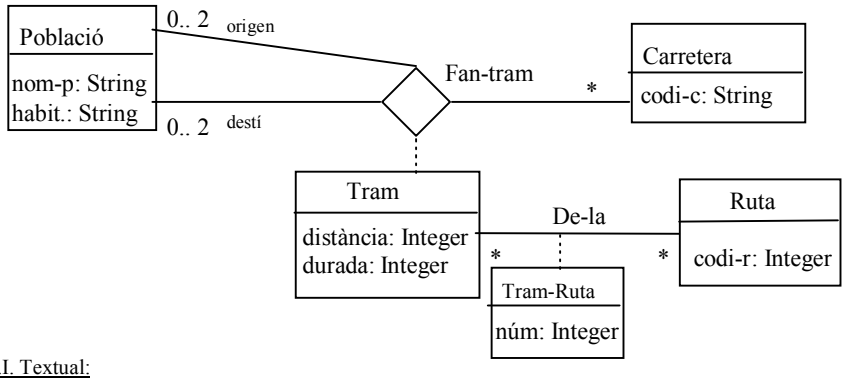
(b) Suposeu que afegim l'operació:

```
obtenirParticipants():Set(String).
```

Aquesta operació ha de retornar el conjunt de noms de les persones que són elegides si és una candidatura amb representació i ha de retornar el conjunt de noms de les persones que es presenten, per la resta de candidatures. Indiqueu a quina o quines classes hem d'assignar aquesta operació i si les operacions són concretes o abstractes. Definiu el o els contractes.

13. Considereu una empresa de transports que està interessada en un sistema per la definició dels recorreguts de les rutes de distribució dels seus camions. L'especificació en UML d'aquest sistema és la següent:

Esquema Conceptual:

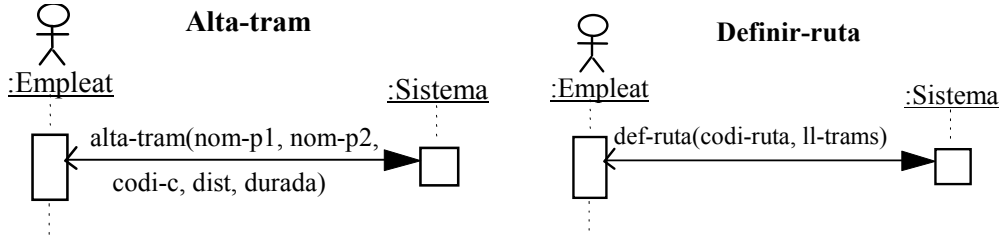


R.I. Textual:

- Claus classes no associatives: (Població, nom-p); (Carretera, codi-c); (Ruta, codi-r)
- La població destí d'un tram d'una ruta ha de coincidir amb la població origen del tram següent (que té núm = núm+1) de la ruta, si el tram següent està definit.
- La població origen i la població destí d'un tram han de ser diferents.
- Una carretera no pot tenir més de 10 trams diferents

Observació: fixeu-vos que, tal i com està definida l'associació *Tram*, donada una carretera, el sistema emmagatzema els trams en les dues direccions possibles. O sigui, el tram (Figueres, La Jonquera, N-II) és diferent de (La Jonquera, Figueres, N-II).

Diagrames de seqüència del sistema i contractes de les operacions:



Contracte de l'operació *alta-tram*:

```
context alta-tram (nom-p1: String, nom-p2: String, codi-c: String,
```

```
                    dist: Enter, durada: Enter)
  -- dóna d'alta un tram entre dues poblacions d'una carretera
pre: 1.1 les poblacions nom-p1 i nom-p2 existeixen
pre: 1.2 la carretera codi-c existeix
pre: 1.3 el tram definit per les poblacions nom-p1 i nom-p2 i la
                carretera codi-c no existeix
post: 2.1 es crea una instància de l'associació fan-tram amb els
     atributs corresponents  on nom-p1 és l'origen i nom-p2 el destí
```

Contracte de l'operació *def-ruta*:

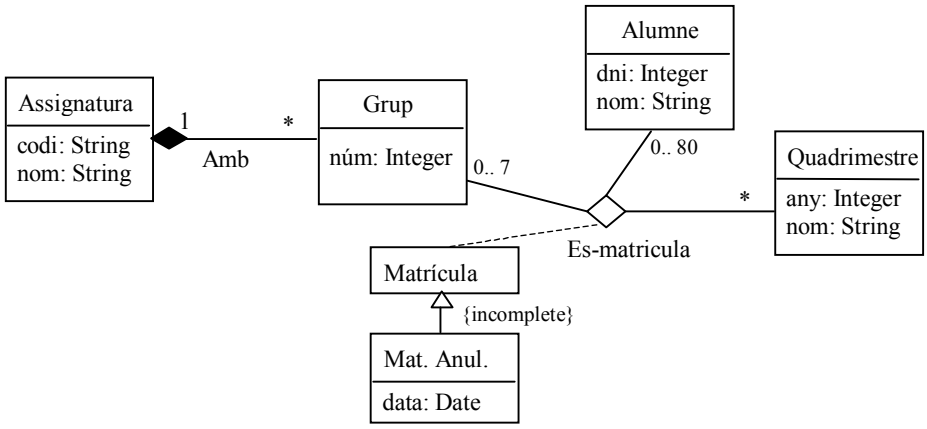
```
context def-ruta (codi-ruta: Enter,
                  ll-trams: Set(String+String+String+Enter))
  -- dóna d'alta una ruta i els trams corresponents
pre: 1.1 la ruta codi-ruta existeix
pre: 1.2 per cada element (nom-p1, nom-p2, codi-c, núm) de ll-trams:
               existeix un tram definit per nom-p1, nom-p2 i codi-c
post: 2.1 per cada element (nom-p1, nom-p2, codi-c, núm) de ll-trams:
             crea una instància de l'associació De-la amb l'atribut
             corresponent
```

Es demana:

a) Diagrama de classes de disseny obtingut com a conseqüència de la normalització de l'esquema conceptual.

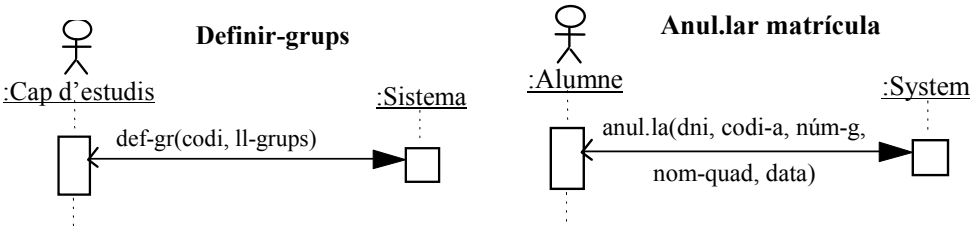
b) Modificació dels contractes de les operacions com a conseqüència de la normalització, si s'escau.

14. Considereu una facultat universitària que està interessada en el disseny d'un sistema software de suport a les matrícules d'assignatures que fan els seus estudiants. L'especificació en UML d'aquest sistema és la següent:



R.I. Textuals:
- Claus classes no associatives: (Assignatura, codi); (Quadrimestre, nom); (Persona, dni)
- Una assignatura no pot tenir dos grups amb el mateix número de grup.
- En un quadrimestre determinat, un alumne no es pot matricular a dos grups d'una mateixa assignatura

Diagrames de seqüència del sistema i contractes de les operacions:



Contracte de l'operació *definir grups*:

```
context def-gr (codi: String, ll-grups: Set(String))
  -- defineix tots els grups de l'assignatura identificada per codi
pre: 1.1 ll-grups no té repetits
pre: 1.2 existeix l'assignatura identificada per codi
pre: 1.3 l'assignatura identificada per codi no té cap grup definit
post: 2.1 per cada element núm-grup que hi ha a ll-grups:
        2.1.1 dóna d'alta un objecte grup amb núm-grup
        2.1.2 crea l'associació entre l'assignatura codi i el grup creat
```

Contracte de l'operació *anul.lar matrícula*:

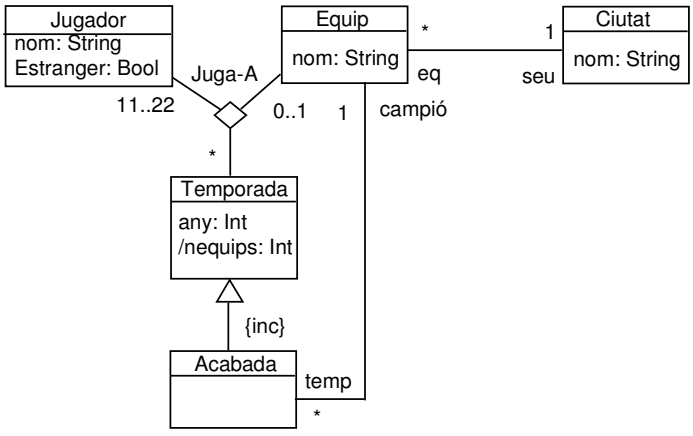
```
context anul.la (dni: Enter, codi: String, núm-g: Enter,
                 nom-quad: String, data: Date)
pre: 1.1 existeix la matrícula no anul.lada de l'alumne dni en el
          grup (núm-g, codi) el quatrimestre nom-quad
post: 2.1 s'especialitza la matrícula com a anul.lada, amb la data
          corresponent
```

Es demana:

a) Diagrama de classes de disseny obtingut com a conseqüència de la normalització de l'esquema conceptual.

b) Modificació dels contractes de les operacions com a conseqüència de la normalització, si s'escau.

15. Sigui el model següent:



RT1: Claus (Jugador, nom), (Equip, nom), (Temporada, any), (Ciutat, nom)
RT2: Un jugador només pot jugar a un màxim de 5 equips al llarg de la seva carrera profesional
RT3: Com a molt, hi ha una instància de Temporada que no està Acabada
ID1: /nequips = nombre d'equips que tenen jugadors que juguen a la temporada

es demana:

a) normalitzeu el model suposant que la informació derivada es materialitza. No cal escriure els tipus dels atributs.

b) sigui l'operació següent:

context Ciutat :: campionsPedrerencs(): Set(String)
post: retorna el llistat de noms d'equips de la ciutat *self* que han sigut campions com a mínim una temporada acabada sense jugar amb estrangers aquella temporada

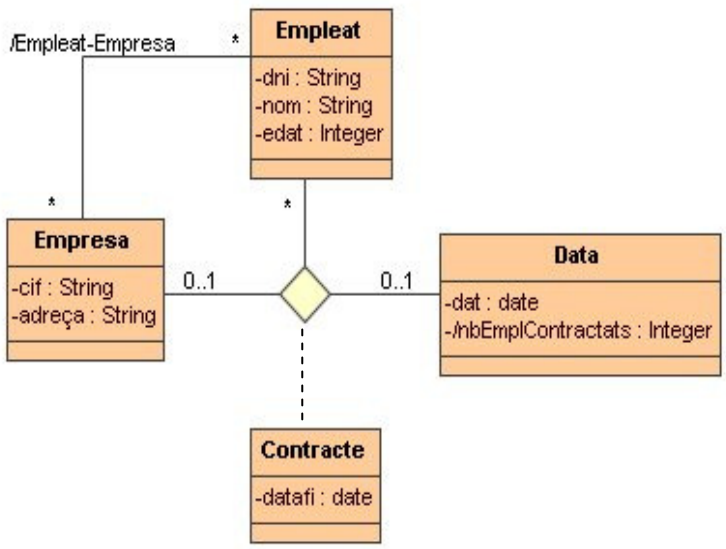
Doneu el seu diagrama de seqüència. Indiqueu les navegabilitats resultants en el model normalitzat i declareu les operacions auxiliars no implícites que us hagin sortir.

c) sigui l'operació següent (amb contracte ja normalitzat):

context Jugador :: retirada()
pre: 1.1 existeix una instància de Temporada no Acabada
pre: 1.2 l'equip del jugador té més d'onze jugadors jugant aquesta Temporada no Acabada
pre: 1.3 el jugador juga a algun equip aquesta Temporada no Acabada
post: 2.1 enregistra que un jugador deixa de jugar en el seu equip la Temporada no Acabada

Doneu el seu diagrama de seqüència donant per suposat que les precondicions són realment precondicions (és a dir, que no cal que les verifiqueu). Afegiu en el model les navegabilitats resultants a les que ja havien aparegut a l'operació anterior, i declareu les operacions auxiliars no implícites que us hagin sortir.

16. Normalitzeu l'esquema conceptual i el contracte següent, suposant que l'associació derivada s'ha de materialitzar i l'atribut derivat s'ha de calcular.

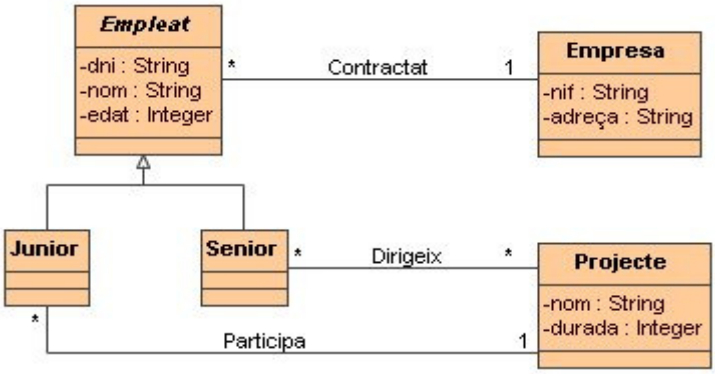


R.Integritat Textuals:
- Claus classes no associatives: (Empresa, cif); (Empleat, dni); (Data, dat)
- Un empleat no pot tenir contractes solapats
- Un empleat no pot tenir més de 10 contractes
- L'atribut datafi de Contracte pot no tenir valor

R.derivació:
- Empleat-Empresa conté els empleats que té o ha tingut contractats una empresa.
- nbEmplContractats indica el nombre d'empleats que s'han contractat en una data per les diverses empreses.

context contractar(cif: String, dni: String,d: Date)
pre: L'empresa cif existeix
pre: L'empleat dni existeix
pre: La data d existeix
pre: No existeix el contracte entre l'empresa cif i l'empleat dni
post: S'ha donat d'alta el contracte entre l'empresa cif, l'empleat dni i la data d

17. Donat l'esquema conceptual i els contractes següents:



R.Integritat Textuals:

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

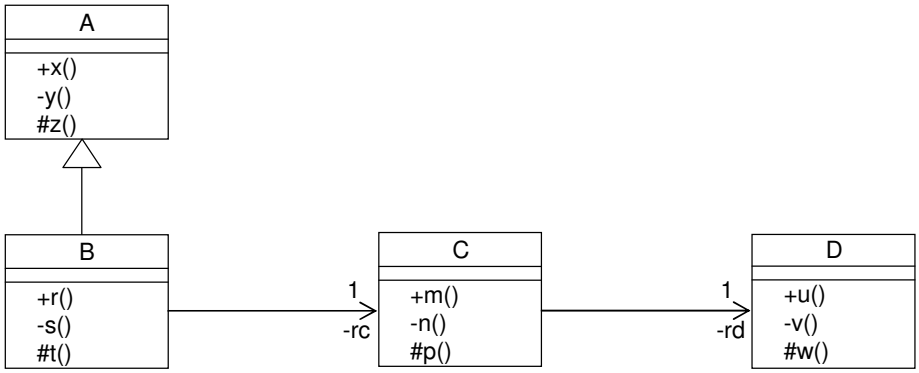
context Empleat::dedicació(): Integer
post resultat = si l'empleat és junior retorna la durada del projecte on participa. Si l'empleat és senior retorna el sumatori de les durades dels projectes que dirigeix

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

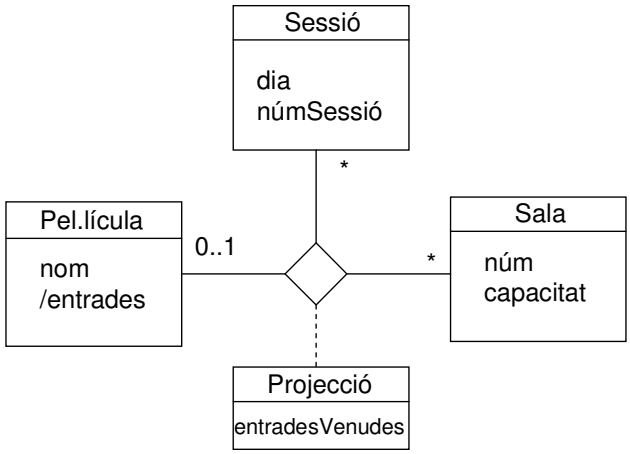
Es demana:

a) Diagrama de seqüència de l'operació *dedicació*.
b) Diagrama de seqüència de l'operació *companysProjecteSenior*.
Indiqueu la navegabilitat resultant del disseny de les operacions anteriors.

18. Sigui el diagrama de classes següent. Sigui l'operació context B::f(d: D). Digues quines invocacions d'operació es poden efectuar en el diagrama de seqüència de *f*. Exemple: una invocació vàlida seria *self.r()*.



19. Normalitzeu el model conceptual de dades i contracte següents que s'usen en un cine multisala:



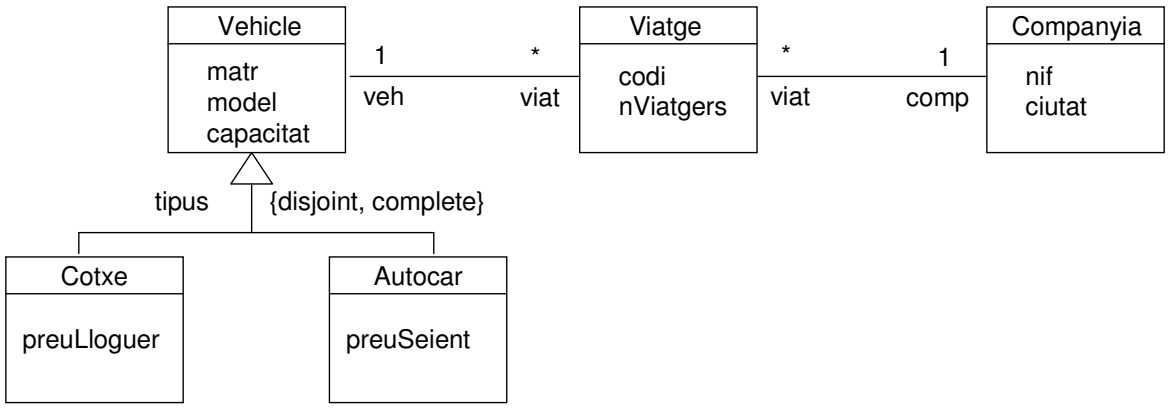
Restriccions d'integritat:
RI1. Claus: Pel.lícula → nom, Sala → núm, Sessió → dia+númSessió
RI2. Una projecció no pot tenir més entrades venudes que la capacitat de la seva sala.
RI3. Una pel.lícula no es pot projectar més de 10.000 vegades

Informació derivada (que cal materialitzar):
ID1. /entrades = suma de les entradesVenudes de les projeccions de la pel.lícula

context novaProjecció(nomP, diaD, numSess, numSal)
pre 1.1: existeixen la pel.lícula nomP, la sessió diaD+numSess, i la sala numSal
pre 1.2: no existeix una projecció entre els tres objectes involucrats
post 2.1: dóna d'alta una projecció entre els tres objectes involucrats

context vendaEntrada(nomP, diaD, numSess, numSal)
pre 1.1: existeix una projecció entre la pel.lícula nomP, la sessió diaD+numSess, i la sala numSal
post 2.1: incrementa l'atribut entradesVenudes de la projecció entre els tres objectes involucrats

20. Sigui el model conceptual de dades de la figura. Es demana que dissenyeu el contracte de les dues operacions que apareixen.



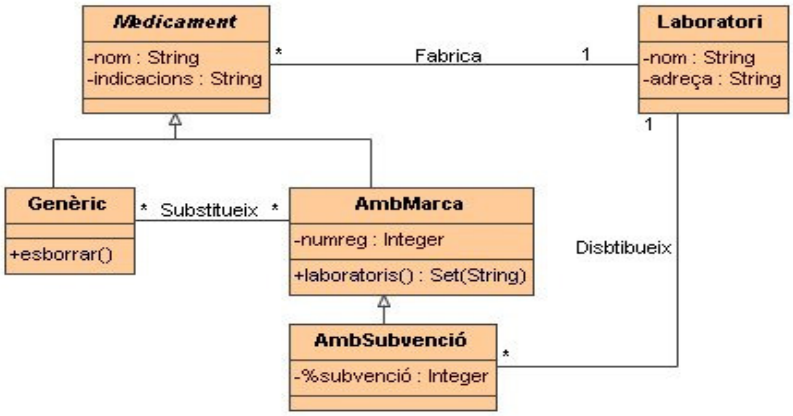
Restriccions d'integritat:
RI1. Claus: Vehicle → matr, Viatge → codi, Companyia → nif
RI2. El nombre de viatgers d'un vehicle no pot ultrapassar la seva capacitat

context Autocar::seientsLliures(): Enter
post: retorna el número de seients lliures de l'autocar

context Companyia::costTransport(): Enter
post: retorna la suma dels preus dels viatges de la companyia tenint en compte que:
- si un viatge s'ha fet per Cotxe, el preu del viatge és igual al preuLloguer del cotxe
- si un viatge s'ha fet per Autocar, el preu del viatge és igual al producte del nViatgers del viatge multiplicat per el preuSeient de l'autocar

Declareu totes les operacions auxiliars que us surtin. Feu també els seus diagrames de seqüència tret de les operacions consultores d'atributs (*getters*). Mostreu clarament les navegabilitats finals i qualsevol altre acoblament que pugui produir-se.

21. 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 AmbMarca :: laboratoris():Set(String)
post result= si *self* és un medicament amb subvenció aleshores es retorna el nom del laboratori que el distribueix. En cas contrari, es retorna el nom dels laboratoris que fabriquen els medicaments genèrics que substitueixen a *self*.

context Genèric :: esborrar()
post s'eliminen les associacions entre *self* i els seus medicaments substituts amb marca.
post s'elimina l'associació entre *self* i el laboratori que el fabrica.
post s'elimina la instància *self*.

Es demana:

a) Diagrama de seqüència de l'operació laboratoris().
b) Indiqueu les navegabilitats resultants del disseny de l'operació laboratoris().
c) A partir de les navegabilitats anteriors, feu el diagrama de seqüència de l'operació esborrar().
d) Afegiu, si cal, les noves navegabilitats resultants del disseny de l'operació esborrar().

22. Considereu el fragment del disseny indicat a la figura següent, corresponent a un sistema que manté la informació dels grups d'assignatures impartides per professors a la nostra universitat. Aquest disseny es basa en diversos supòsits que els usuaris consideren acceptables ara per ara, però no descarten que en el futurs calgui ampliar-los. Es demana:

(a) El sistema permet donar d'alta nous professors a la Universitat. Per fer-ho, el contracte que assigna el codi a un nou professor és el següent:

```
context Professor :: professor (nomp: String)
pre: no existeix el professor nomP
post: es crea una nova instància de Professor amb nomP i codi =
      darrer codi assignat a algun professor + 1
```

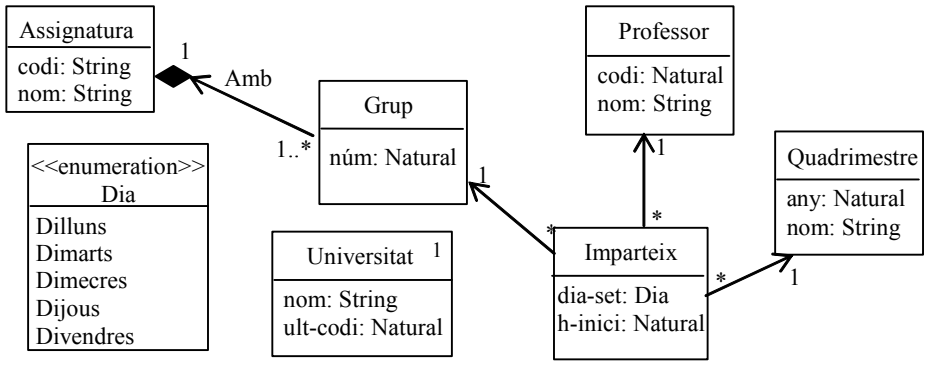
Es vol canviar el sistema de generació de codis que són assignats als professors, sense canviar el tipus de l'atribut. No es tracta d'incorporar el canvi ara (que tampoc sabem exactament com serà), sinó que es tracta que si en el futur es vol incorporar, el sistema només s'hagi d'estendre, sense modificar-lo. Critica el fragment del contracte que assigna el codi a un nou empleat i proposa una forma alternativa d'assignació d'aquests codis que minimitzi l'impacte del canvi.

(b) El sistema suposa que els professors imparteixen els grups de les assignatures. En el futur, està previst que els becaris també puguin impartir classes. Modifiqueu el disseny, de manera que el sistema estigui protegit contra aquest canvi. No es tracta d'incorporar el canvi ara (que potser no es faria servir mai), sinó que es tracta que si en el futur es vol incorporar, el sistema només s'hagi d'estendre, sense modificar-lo.

(c) Expliqueu què s'haurà de fer, en el futur i respecte del disseny de l'apartat (b), si es vol que els becaris també imparteixin grups d'assignatures.

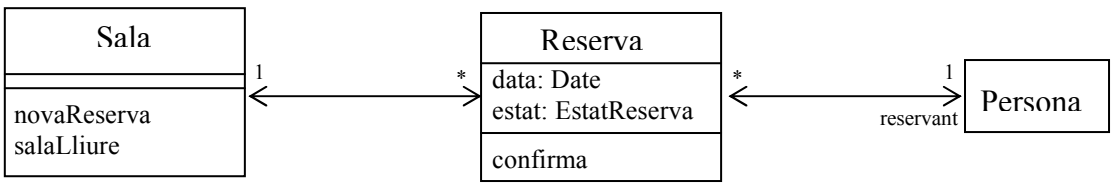
(d) El sistema suposa que l'horari en que s'imparteix un grup d'una assignatura és un dia a la setmana amb un període de dues hores consecutives. En el futur, l'horari podria ser dos dies a la setmana amb o sense la mateixa hora d'inici o tres dies a la setmana o fins i tot els cinc dies. L'exercici consisteix a modificar el disseny, de manera que el sistema estigui protegit contra els canvis indicats. Indiqueu les modificacions en els contractes de les operacions existents, i definiu també els contractes de les noves operacions que afegiu (si és el cas). No es tracta d'incorporar els canvis ara (que potser no es farien servir mai), sinó que es tracta que si en el futur es vol incorporar algun dels canvis esmentats, el sistema només s'hagi d'estendre, sense modificar-lo.

(e) Expliqueu què s'haurà de fer, en el futur i respecte del disseny de l'apartat (d), si es vol que l'horari sigui per a un conjunt qualsevol de dies amb o sense la mateixa hora d'inici.




```
context Assignatura
inv codi-és-clau: no hi ha dues assignatures amb el mateix codi

context Quadrimestre
inv nom-és-clau: no hi ha dos quatrimestres amb el mateix nom

context Grup
inv núm-i-codi-són-clau: no hi ha dos grups amb el mateix núm en la
                         mateixa assignatura

context Imparteix
inv imparteix-és-ternària: no hi ha dues instàncies d'imparteix
           relacionades amb la mateixa terna grup-professor-quatrimestre

context Professor
inv no-hi-ha-solapaments: un professor no pot tenir dues imparticions el
      mateix quadrimestre, el mateix dia, amb dues hores d'inici
      separades per menys de dues hores

context Assignatura :: assignacio_docent (p: Professor, q: Quadrimestre,
                                          g: Grup, d: Dia, hi: Natural)
pre grup-d-assignatura: g és grup de self
post nova-impartició: crea una instància d'Imparteix relacionada amb p,
                      q i g, amb dia-set = d i h-inici = hi
```

23. Considereu el fragment del disseny indicat a la figura següent, corresponent a un sistema de reserva de sales que estem dissenyant. Aquest disseny es basa en diversos supòsits que els usuaris consideren acceptables ara per ara, però no descarten que en el futurs calgui ampliar-los.




```
context Sala::novaReserva(qui: Persona, quan: Date): EstatReserva
  pre no-caducada: quan >= CurrentDate - CurrentDate: data del sistema
  post nova-reserva: crea una nova Reserva amb data = quan, reservant =
       qui, lligada amb la sala self, complint que:
            if salaLliure(quan) then r.estat = EstatReserva::Confirmada
            else r.estat = EstatReserva::EnEspera
            endif
  post resultat: result = r.estat

context Sala::salaLliure(quan: Date): Boolean
  post: result = la sala self no té cap reserva amb data = quan

context Reserva
  inv no-solapament-reserves: no hi ha dues reserves confirmades a la
                              mateixa data
```

```
context Reserva::confirma()
  pre sala-lliure: sala.salaLliure(data)
  pre reserva-en-espera: estat = EstatReserva::EnEspera
  post confirma-reserva: estat = EstatReserva::Confirmada
```

(a) El sistema suposa que el reservant és sempre una persona. En el futur, el reservant podria ser també una Empresa. Modifiqueu el disseny, de manera que el sistema estigui protegit contra aquest canvi. No es tracta d'incorporar el canvi ara (que potser no es faria servir mai), sinó que es tracta que si en el futur es vol incorporar, el sistema només s'hagi d'estendre, sense modificar-lo.

(b) Expliqueu què s'haurà de fer, en el futur i respecte del disseny de l'apartat (a), si es vol que el reservant pugui ser també una Empresa.

(c) El sistema suposa que la reserva és per a una sola sala. En el futur, una reserva podria ser d'un conjunt de sales (per exemple, per a fer un congrés). No es voldria fer diverses reserves, cadascuna d'una sola sala, sinó que una reserva pogués ser de més d'una sala. Modifiqueu el disseny, de manera que el sistema estigui protegit contra el canvi indicat. No es tracta d'incorporar el canvi ara (que potser no es faria servir mai), sinó que es tracta que si en el futur es vol incorporar, el sistema només s'hagi d'estendre, sense modificar-lo.

(d) Expliqueu què s'haurà de fer, en el futur i respecte del disseny de l'apartat (c), si es vol que una reserva pugui ser de diverses sales.

(e) El sistema suposa que les reserves són per a un sol dia. Els usuaris consideren aquest supòsit acceptable ara per ara, però no es descarta que en el futur una reserva pugui ser pels dies compresos entre un dia inicial i un dia final, per una setmana, per un conjunt de dies no consecutius, per uns dies amb una certa periodicitat (p.e . els dimarts) etc. No es contempla que una reserva pugui ser per unitats diferents de dia (no hores, no mesos, etc.). L'exercici consisteix a modificar el disseny, de manera que el sistema estigui protegit contra els canvis indicats. Indiqueu les modificacions en els contractes de les operacions existents, i definiu també els contractes de les noves operacions que afegiu (si és el cas). No es tracta d'incorporar els canvis ara (que potser no es farien servir mai), sinó que es tracta que si en el futur es vol incorporar algun dels canvis esmentats, el sistema només s'hagi d'estendre, sense modificar-lo.

(f) Expliqueu què s'haurà de fer, en el futur i respecte del disseny de l'apartat (e), si es vol que una reserva pugui per a un conjunt qualsevol de dies, no necessàriament consecutius.