Introducció a l'Enginyeria del Programari

Capítol 3. Disseny

Enunciats exercicis

PART I. EXERCICIS QUE COBREIXEN TOT EL PROCÉS DE DISSENY

Exercici 1. .Laboratoris farmacèutics

Apartat 1. Una agrupació de laboratoris farmacèutics ens ha demanat que dissenyem una primera versió 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. La informació que es guarda dels medicaments és el seu nom (que l'identifica) i les seves indicacions (és a dir, la seva recepta) i el laboratori que el fabrica. Del laboratori es guarda el seu nom (també aquí és un identificador) i la seva adreça.

Considereu el model de dades següents, i tres casos d'ús essencials amb el model del comportament associat:

cas d'ús Nou Medicament

activació Un laboratori farmacèutic vol donar d'alta un nou medicament

escenari principal El Laboratori Farmacèutic entra el nom i les indicacions d'un medicament. El Sistema enregistra aquest nou medicament amb referència al Laboratori Farmacèutic com a fabricant.

escenaris alternatius

Ja existeix un medicament amb aquest nom: El Sistema demana un nom alternatiu, donant l'opció al Laboratori Farmacèutic de finalitzar el cas d'ús

cas d'ús Consultar Medicament

activació L'agrupació de laboratoris farmacèutics vol conèixer la informació d'un medicament

escenari principal L'Agrupació entra el nom d'un medicament. El Sistema mostra el nom del laboratori farmacèutic que el fabrica, i les seves indicacions.

escenaris alternatius

No existeix un medicament amb aquest nom: El Sistema demana un nom alternatiu, donant l'opció a l'Agrupació de finalitzar el cas d'ús

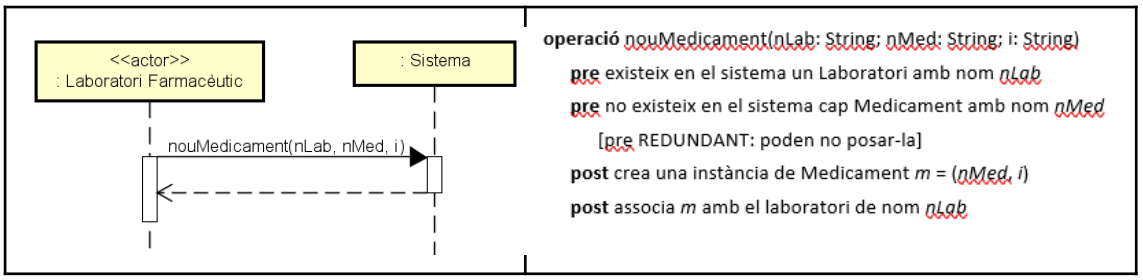
cas d'ús Medicaments de Laboratori

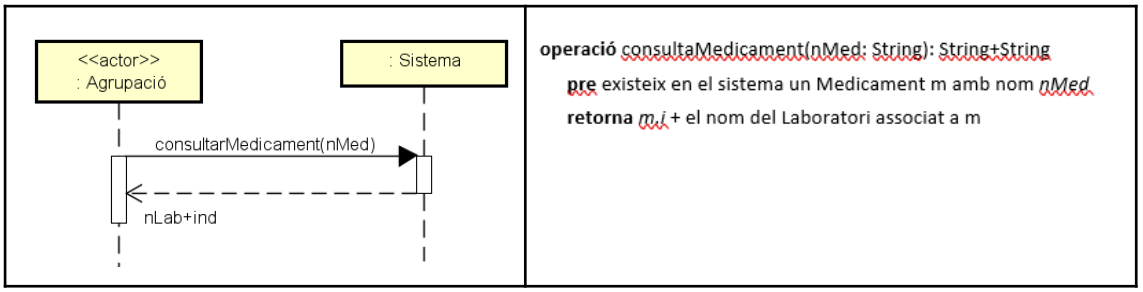
activació L'agrupació de laboratoris farmacèutics vol consultar la llista de medicaments que fabrica un laboratori

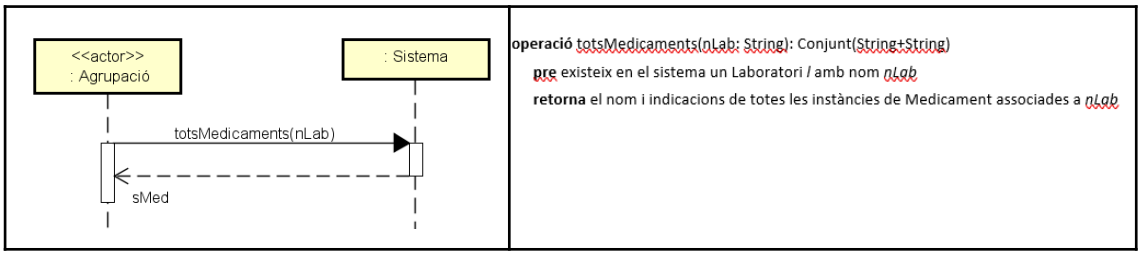
escenari principal L'Agrupació entra el nom d'un laboratori farmacèutic. El Sistema mostra la llista de medicaments (el seu nom i les seves indicacions) que fabrica el laboratori.

escenaris alternatius

No existeix un laboratori farmacèutic amb aquest nom: El Sistema demana un nom alternatiu, donant l'opció a l'Agrupació de finalitzar el cas d'ús



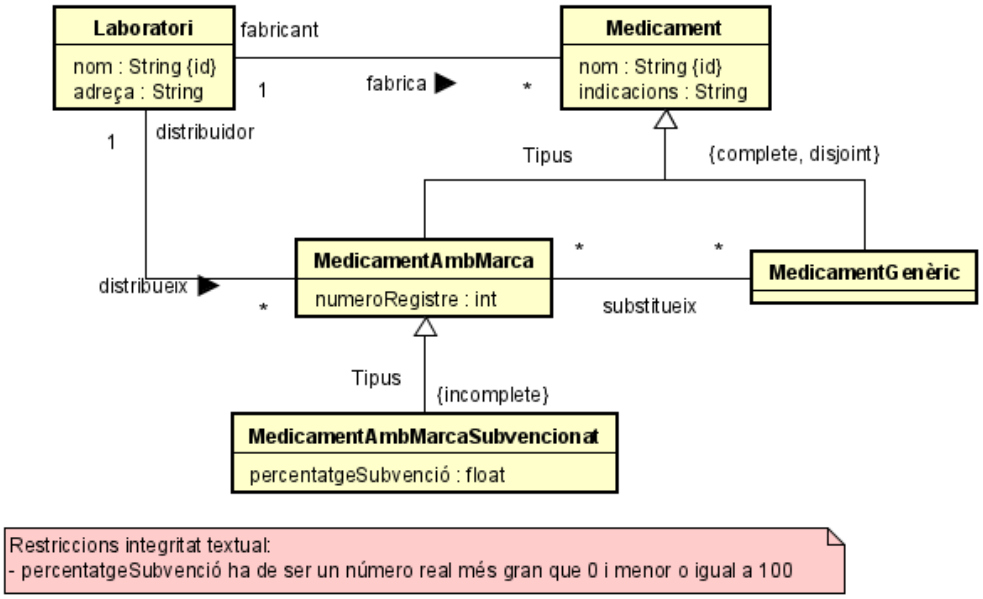




Es demana que feu un disseny complet de la solució, aplicant els quatre passos habituals.

Apartat 2. Una vegada entregada la primera versió, l'agrupació de laboratoris farmacèutics ens indica que els medicaments que fabriquen poden ser genèrics o amb marca. Dels medicaments amb marca s'ha de guardar el seu número de registre. Cada medicament genèric pot ser substituït per uns certs medicaments amb marca i vice-versa. Alguns medicaments amb marca són subvencionats per la seguretat social; d'aquests últims hem de guardar el percentatge de subvenció que se'ls li aplica. Eventualment, també, els medicaments amb marca poden ser distribuïts per un laboratoris de l'agrupació diferent del laboratori de fabricació.

Aquesta extensió de l'enunciat es reflecteix en un model de dades nou, i també dóna lloc a un nou cas d'ús essencial amb el seu model del comportament:



cas d'ús Nou Medicament Amb Marca

activació Un laboratori farmacèutic vol donar d'alta un nou medicament amb marca

escenari principal El Laboratori Farmacèutic entra el nom, les indicacions i el número de registre d'un medicament amb marca. Si s'escau, també entra el percentatge de subvenció atorgat per la seguretat social i el laboratori de distribució i, si és el cas, tots aquells medicaments genèrics que poden substituir el medicament amb marca que estem donant d'alta. El Sistema enregistra aquest nou medicament amb marca, amb referència al Laboratori Farmacèutic com a fabricant i relacionant amb tots els medicaments genèrics i amb el laboratori de distribució, quan és el cas.

escenaris alternatius

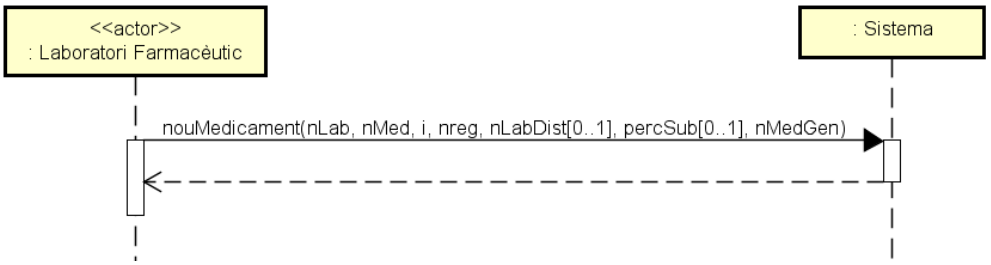
Ja existeix un medicament amb aquest nom: El Sistema demana un nom alternatiu, donant l'opció al Laboratori Farmacèutic de finalitzar el cas d'ús

Percentatge de subvenció incorrecte (fora de l'interval (0, 100]): El Sistema avisa que el percentatge no és correcte i demana un de nou

No existeix el laboratori de distribució: El Sistema demana un nom alternatiu, donant l'opció al Laboratori Farmacèutic de finalitzar el cas d'ús

El laboratori de distribució és el mateix que el laboratori de fabricació: El Sistema demana un nom alternatiu, donant l'opció al Laboratori Farmacèutic d'entrar un altre nom o fins i tot de finalitzar el cas d'ús

No existeix un medicament genèric referenciat com a substitut del medicament amb marca: El Sistema demana un nom alternatiu, donant l'opció al Laboratori Farmacèutic de finalitzar el cas d'ús, donant l'opció al Laboratori Farmacèutic d'entrar un altre nom o fins i tot de finalitzar el cas d'ús



operació nouMedicamentAmbMarca(nLab: String; nMed: String; i: String; nreg: Integer;
nLabDist: String[0..1]; percSub: Float[0..1]; nMedGen: Conjunt(String))
-- El conjunt no cal dir que és opcional, simplement pot ser buit

pre existeix en el sistema un Laboratori amb nom *nLab*

pre no existeix en el sistema cap Medicament amb nom *nMed* [pre REDUNDANT: poden no posar-la]

pre si *nLabDist* no és nul, aleshores *nLabDist* ≠ *nLab* AND existeix un Laboratori *l2* amb nom *nLabDist*

pre si *percSub* no és nul, aleshores 0 < *percSub* ≤ 100 [pre REDUNDANT: poden no posar-la]

pre per tot element *mg* dins de *nMedGen*: existeix en el sistema un MedicamentGenèric amb nom *mg*

post si *percSub* no és nul, crea una instància de MedicamentSubvencionat *m* = (*nMed*, *i*, *nreg*, *percSub*)

post si *percSub* és nul, crea una instància de MedicamentAmbMarca *m* = (*nMed*, *i*, *nreg*)

post associa *m* amb el laboratori de nom *nLab*

post si *nLabDist* no és nul, associa *m* amb el laboratori de nom *nLabDist*

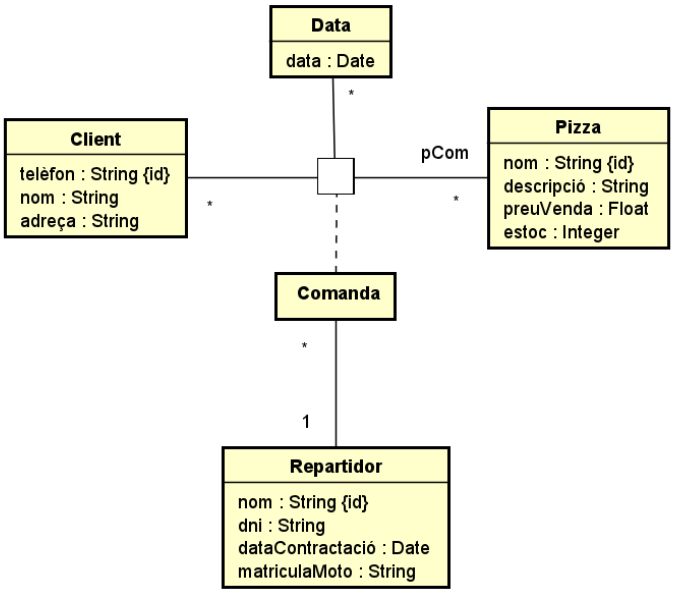
post per tot element *mg* dins de *nMedGen*, associa m amb el MedicamentGenèric amb nom *mg*

Es demana el disseny d'aquest nou cas d'ús sobre la solució del primer apartat.

Exercici 2. Comandes pizzeria

Una pizzeria ens ha demanat que dissenyem un sistema informàtic per gestionar la venda de pizzes per telèfon. Dels clients registrats es guarda el nom, adreça i telèfon (que l'identifica; podeu assumir que el sistema de la pizzeria pot saber el número del telèfon que truca). Les pizzes tenen un nom que les identifica, la descripció, el preu de venda al públic i l'estoc. Les comandes les entreguen els repartidors de la pizzeria, dels qual es guarda el nom, dni, la data en que van ser contractats i la matrícula de la moto que usaran (que és propietat de la pizzeria). El repartidor s'assigna en el mateix moment de concretar la comanda. La pizzeria assumeix que un repartidor completarà una comanda en 30 minuts com a molt, i per això sempre reserva 30 minuts com a temps de servei de la pizza (independentment que al final el repartidor pugui anar més ràpid). Per a totes les variants que venen a continuació, es dona un model de dades, i un conjunt casos d'ús essencials (només l'escenari principal) amb el model del comportament associat. En tots els apartats, es demana que feu un disseny complet de la solució.

Apartat 1. En la primera entrega del sistema, un client només pot demanar una pizza en cada comanda:

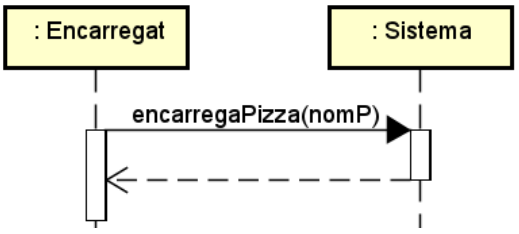


Restriccions integritat textuals:
RIT1. Els atriburs dni corresponen a un DNI correcte
RIT2. Els atributs enters i reals han de ser positius, l'atribut estoc pot ser 0
RIT3 (r: Repartidor): La data de contractació de r es menor o igual a la data de les comandes que entrega
RIT4 (r: Repartidor): La data/hora entre totes les comandes entregades per r tenen com a mínim 30 minuts de diferència

cas d'ús Encarrega Pizza

activació Un Client truca per encarregar una pizza

escenari principal El Client comunica a l'Encarregat de la pizzeria el nom de la pizza que vol encarregar. L'Encarregat entra aquesta dada i el Sistema enregistra la comanda (dia, hora i minuts) actual, i li assigna un repartidor disponible

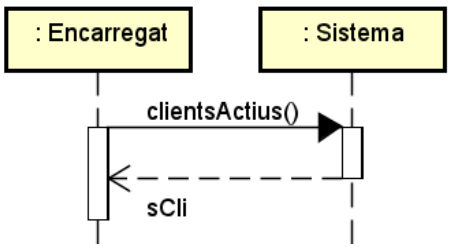


operació encarregaPizza(nomP: String)
pre existeix una pizza p amb p.nom = nomP
post crea una instància c de Comanda associant p amb el client que fa la trucada, i amb la data actual
post associa c amb un repartidor r disponible per entregar la comanda

cas d'ús Clients Actius

activació L'Encarregat de la pizzeria vol saber quins clients han encarregat una pizza en la darrera setmana

escenari principal L'Encarregat demana al Sistema quins clients han encarregat una pizza en la darrera setmana, i el Sistema respon amb el nom i telèfon de tots aquests clients

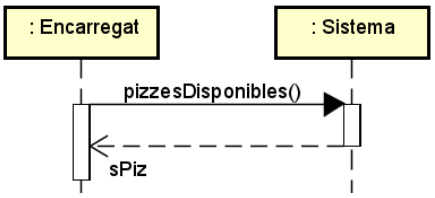


operació clientsActius(): Set(Str.+Str.)
retorna el conjunt sCli que conté c.nom i c.telèfon de tots els Clients c que han encarregat una pizza en la darrera setmana

cas d'ús Pizzes Disponibles

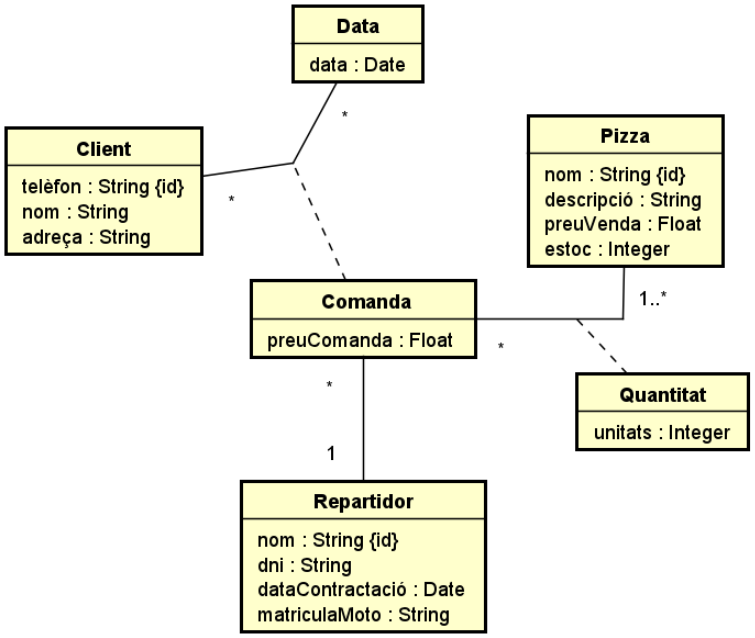
activació L'Encarregat de la pizzeria vol saber quines pizzes tenen estoc

escenari principal L'Encarregat demana al Sistema quines pizzes tenen actualment estoc, i el Sistema respon amb el nom d'aquestes pizzes

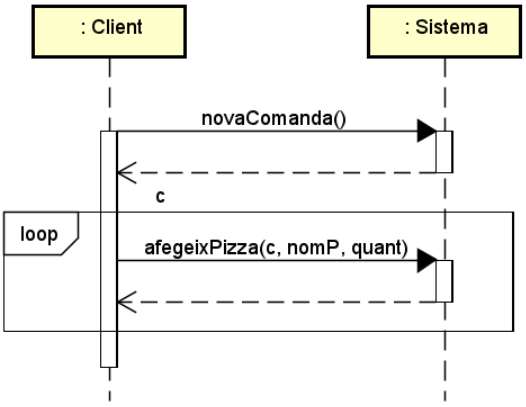


operació pizzesDisponibles(): Set(String)
retorna el conjunt sPiz amb els noms de totes les Pizzes p tal que p.estoc > 0

Apartat 2. Una vegada tenim en marxa el sistema, ens diuen que els clients poden demanar tantes pizzes com vulguin a cada comanda. Per cada comanda s'ha de guardar el preu total. Això porta a la modificació següent dels models d'especificació:



Restriccions integritat textuals:
RIT1. Els atriburs dni corresponen a un DNI correcte
RIT2. Els atributs enters i reals han de ser positius, l'atribut estoc pot ser 0
RIT3 (r: Repartidor): La data de contractació de r es menor o igual a la data de les comandes que entrega
RIT4 (r: Repartidor): La data/hora entre totes les comandes entregades per r tenen com a mínim 30 minuts de diferència

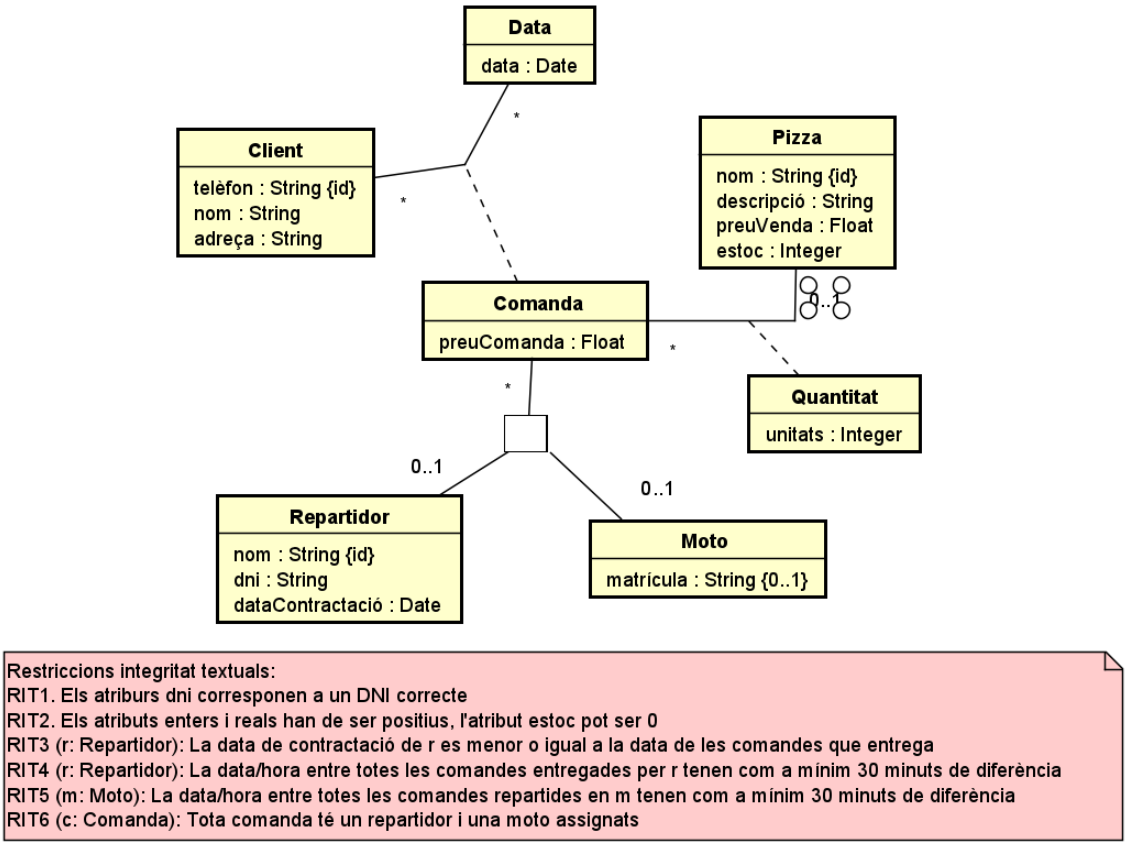



```
operació novaComanda(): Comanda
pre existeix un Repartidor disponible
      per entregar la comanda
post crea una instància c de Comanda
      entre el client que fa la trucada i
      la data actual
post c.preuComanda = 0
post associa c amb un Repartidor
      disponible per entregar la comanda
retorna c

operació afegeixPizza
     (c: Comanda; nomP: String;
      quant: Integer)
pre existeix una pizza p amb p.nom = nomP
pre p.estoc >= quant
post crea una instància q de Quantitat
      entre c i la pizza p, amb q.unitats
      = quant
post c.preuComanda += p.preuVenda*quant
post p.estoc -= quant
```

Es demana que disseny aquest nou cas d'ús sobre la solució del primer apartat.

Apartat 3. A més, per minimitzar el número de motos emprades, les motos no estan assignades a un repartidor concret, sinó que s'assignarà al repartidor per una comanda concreta, en el mateix moment de fer una comanda. El nou diagrama de classes és:



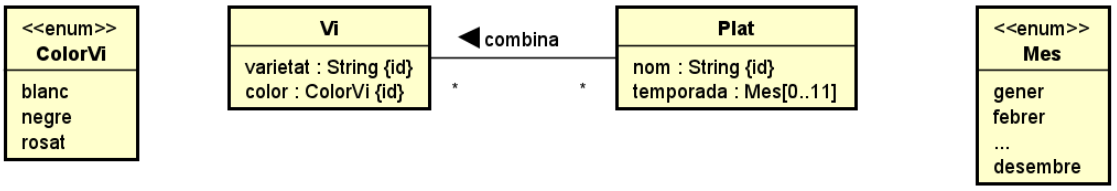
Restriccions integritat textuals:
RIT1. Els atriburs dni corresponen a un DNI correcte
RIT2. Els atributs enters i reals han de ser positius, l'atribut estoc pot ser 0
RIT3 (r: Repartidor): La data de contractació de r es menor o igual a la data de les comandes que entrega
RIT4 (r: Repartidor): La data/hora entre totes les comandes entregades per r tenen com a mínim 30 minuts de diferència
RIT5 (m: Moto): La data/hora entre totes les comandes repartides en m tenen com a mínim 30 minuts de diferència
RIT6 (c: Comanda): Tota comanda té un repartidor i una moto assignats

mentre que cal afegir una pre al primer contracte:

```
operació novaComanda(): Comanda
pre existeix un Repartidor disponible per entregar la comanda
pre existeix una Moto disponible per ser usada pel repartidor
post crea una instància c de Comanda entre el client que fa la trucada i la data
     actual
post c.preuComanda = 0
post associa c amb un Repartidor i una Moto disponibles per entregar la comanda
retorna c
```

Exercici 3. Exquisideses gastronòmiques

Apartat 1. Un reputat gastrònom que està perdent la memòria, ha decidit crear un sistema informàtic per saber quins plats combinen amb quins vins i evitar disbarats com ara servir un Chardonnay blanc acompanyant un entrecot de bou. La informació que vol saber dels vins és la varietat de raïm (p.e., Chardonnay) i el color (blanc, negre o rosat), mentre que dels plats en vol saber el seu nom (com ara entrecot al foie amb reducció Pedro Ximenes), la seva dificultat d'elaboració (fàcil, normal, elaborat) i si és de temporada o no (en cas de ser de temporada, indicant en quins mesos es pot cuinar). Els plats s'identifiquen pel seu nom, mentre que en els vins, cal anar amb compte que una mateixa denominació pot tenir raïm de dos colors diferents. Considereu el següent model de dades d'especificació, i dos casos d'ús amb model del comportament associat:



cas d'ús Combina

activació El Gastrònom afegeix una nova combinació de plat i vins

escenari principal El Gastrònom entra el nom d'un plat i la varietat i color d'un vi. El Sistema enregistra aquesta combinació.

escenaris alternatius

No existeix un plat amb aquest nom: El Sistema demana un nom alternatiu, donant l'opció al Gastrònom de finalitzar el cas d'ús

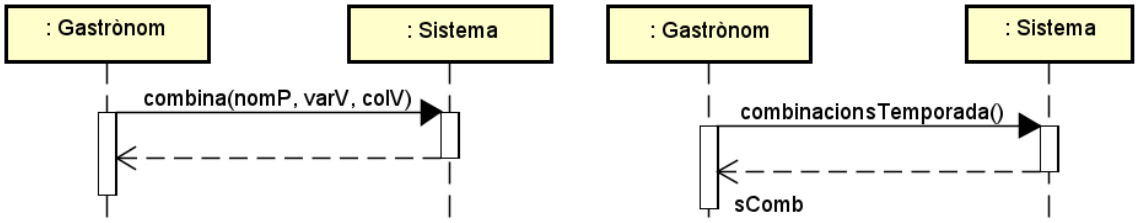
No existeix un vi amb aquesta varietat i color: El Sistema demana una varietat i color alternatius, donant l'opció al Gastrònom de finalitzar el cas d'ús

La combinació ja existeix: El Sistema avisa d'aquesta circumstància i finalitza el cas d'ús

cas d'ús Combinacions de Temporada

activació El Gastrònom vol saber les combinacions de plats i vins pel mes actual

escenari principal El Gastrònom demana al Sistema les combinacions enregistrades el mes actual. El Sistema respon mostrant la llista de plats la temporada dels quals inclou el mes actual, i per cadascun d'aquests plats, la llista de vins que hi combinen



operació combina(nomP: String; varV: String; colV: ColorVi)

pre existeix un plat p amb p.nom = nomP

pre existeix un vi v amb v.varietat = varV i v.color = colV

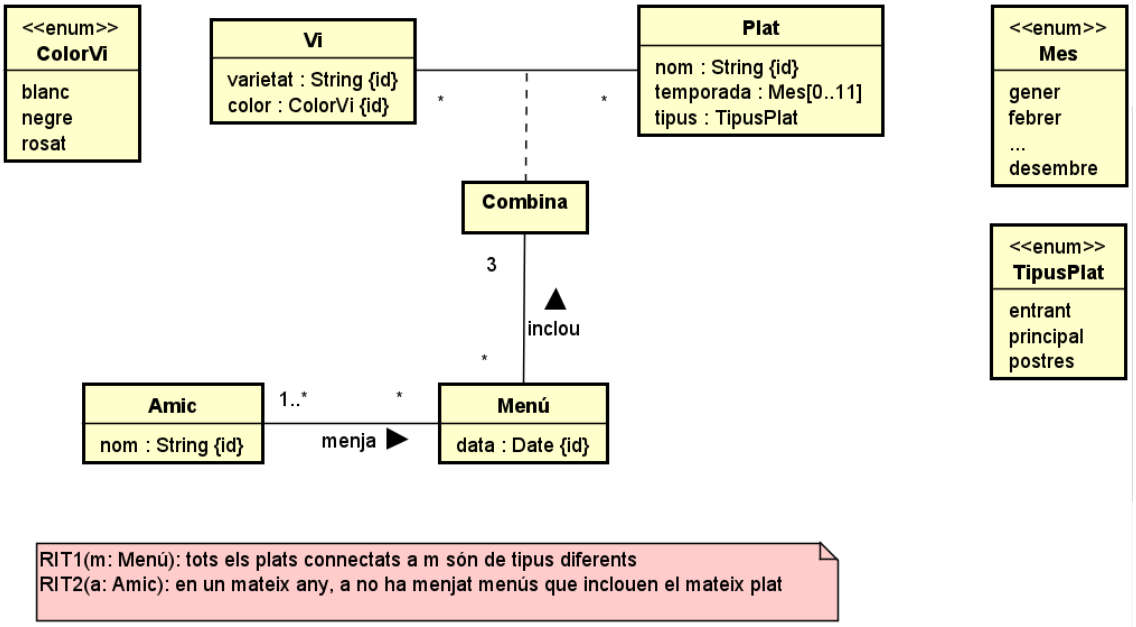
post es crea una associació entre p i v

operació combinacionsTemporada(): Conjunt(Plat + Conjunt(Vi))

retorna el conjunt de plats p tals que p.temporada inclou el mes actual, i per a cada p, el conjunt de vins que combinen amb p)

Es demana que feu un disseny complet de la solució, aplicant els quatre passos habituals.

Apartat 2. El nostre gastrònom favorit s'ha comprat una casa nova amb un pati molt gran i ha decidit anar convidant a sopar els seus nombrosos amics, dels que només sap el seu nom. Com a bon mediterrani, els seus menús es composaran sempre d'un entrant, un plat principal i unes postres (un plat pot ser d'un tipus en un menú, i d'un altre, en un altre menú), tots ells regats amb vins normalment diferents (faltaria més!) i que òbviament han de maridar bé amb el plat corresponent. Degut a la seva ja esmentada mala memòria, el gastrònom vol anar enregistrant a quins amics va oferir quins plats i així evitar la vergonya de servir el mateix plat dues vegades al mateix amic, si més no en el darrer any. Com el gastrònom ja s'ha fet gran i es cansa, no pot oferir més d'un menú la mateixa nit. Aquesta evolució de les necessitats del nostre amic porta a uns canvis en el model de dades i la introducció d'un nou cas d'ús amb el seu model del comportament associat:



cas d'ús Configura Menú

activació El gastrònom vol preparar el menú per al següent sopar

escenari principal El Gastrònom entra el nom dels amics als què convidarà el proper sopar. El Sistema mostra els plats que aquests amics no han menjat en el darrer any, classificats per entrant, plat principal i postres. A continuació, el Gastrònom configura un menú de plats i vins que combinen, que no conté cap dels plats mostrats, i l'entra al Sistema juntament amb la data del sopar. El Sistema enregistra el menú i el relaciona amb els amics convidats

escenaris alternatius

No existeix algun dels amics: El Sistema demana un nom alternatiu, donant l'opció de donar per acabada l'entrada d'amics o fins i tot de finalitzar el cas d'ús

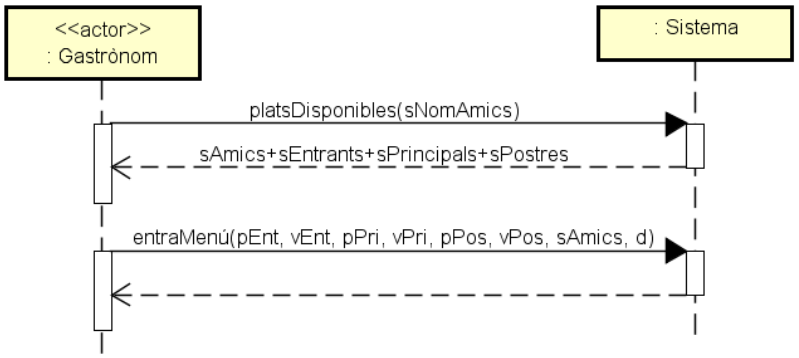
No hi ha plats d'algun tipus no menjats el darrer any: El Sistema finalitza el cas d'ús

El menú inclou plats menjats el darrer any: El Sistema dóna l'opció d'entrar un menú diferent o fins i tot de finalitzar el cas d'ús

El menú inclou més d'un plat del darrer tipus: El Sistema dóna l'opció d'entrar un menú diferent o fins i tot de finalitzar el cas d'ús

El menú inclou un parell de plat i vi que no combinen: El Sistema dóna l'opció d'entrar un menú diferent o fins i tot de finalitzar el cas d'ús

Data de sopar anterior a l'actual: El Sistema demana una nova data



operació platsDisponibles(sNomAmics: Conjunt(String)):
Conjunt(Amic)+Conjunt(String)+Conjunt(String)+Conjunt(String)

pre per a tot *sa* dins *sNomAmics*, existeix en el sistema un Amic amb nom *sa*

pre existeix com a mínim un plat de cada mena (entrant, principal, postre) que no ha estat menjat per cap dels amics de *sNomAmics* en el darrer any

retorna el conjunt d'amics que tenen els noms de *sNomAmics*

retorna el conjunt de noms de plats entrants que no ha estat menjat per cap dels amics de *sNomAmics* en el darrer any

retorna ídem per plats principals

retorna ídem per plats de postres

operació entraMenú(pEnt: String, vEnt: String, pPri: String, vPri: String, pPos: String, vPos: String, sAmics: Conjunt(Amic), d: Date)

pre *pEnt* és un plat entrant, *pPri* és un plat principal, i *pPos* és un plat de postres

pre *pEnt* combina amb *vEnt*, *pPri* combina amb *vPri*, i *pPos* combina amb *vPos*

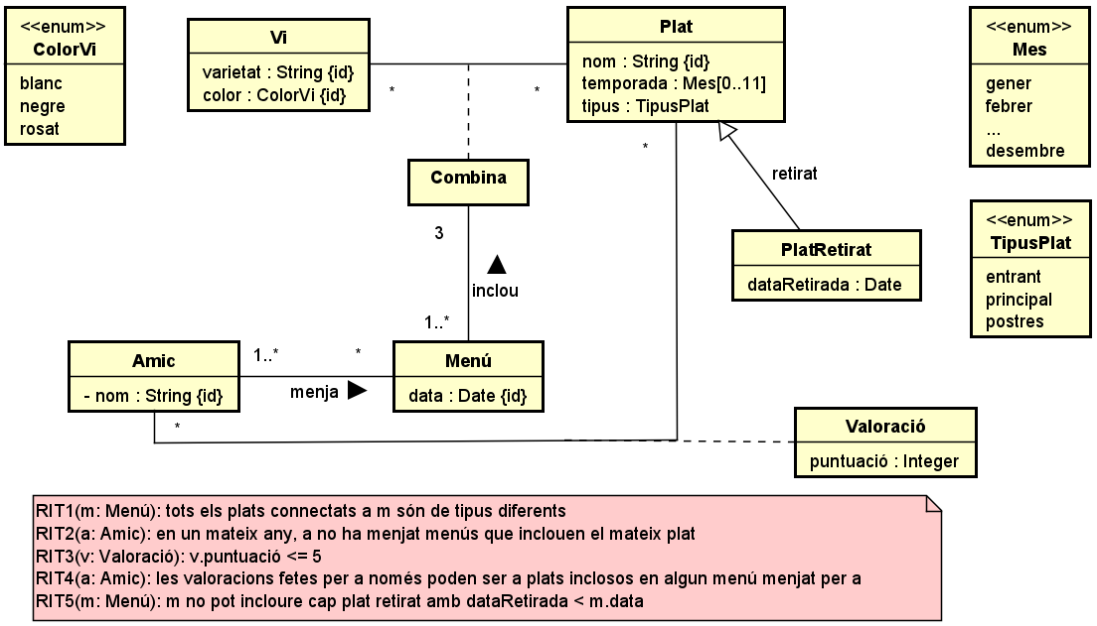
pre cap dels amics en *sAmics* ha menjat *pEnt*, ni *pPri*, ni *pPps*, en el darrer any

post existeix *m*: *Menú* amb data *d* que inclou *pEnt*+*vEnt*, *pPri*+*vPri* i *pPos*+*vPos*

post per tot *a* dins de *sAmics*, *a* menja *m*

Es demana que feu un disseny complet de la solució, aplicant els quatre passos habituals.

Apartat 3. Degut a certes crítiques (segurament injustes) a alguns dels seus menús, el nostre afamat gastrònom ha decidit demanar la valoració dels seus plats als convidats un cop acabat el sopar, amb una puntuació de 1 a 5. Els convidats no estan obligats a donar la seva opinió, i tampoc poden valorar el mateix plat més d'una vegada (fins i tot en sopars diferents). El gastrònom vol tenir una funcionalitat per retirar plats quan considera que les valoracions són massa baixes. Igualment, tenim alguns canvis en el model d'especificació:



cas d'ús Valorar Plats

activació Els comensals ja han acabat de menjar i és hora de tocar el dos

escenari principal Els Comensals que volen entren la seva valoració dels plats en el terminal que el Gastrònom ha posat a la seva disposició en el rebedor. El Sistema enregistra les valoracions.

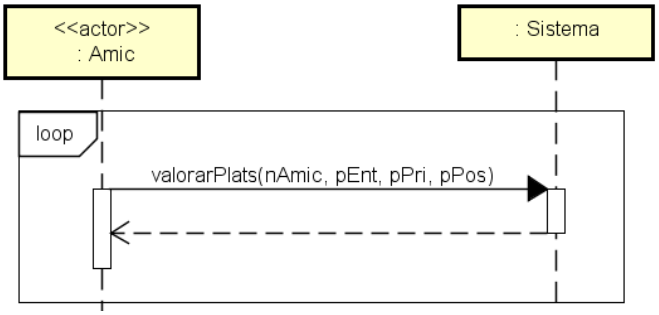
escenaris alternatius

No existeix algun dels amics: El Sistema demana al Comensal el seu nom correcte, donant l'opció que no entri puntuació

Amic ja ha puntuat: El Sistema avisa que algú amb el nom donat ja ha entrat una puntuació per a un plat del menú, demanat que puntuï un altre plat o fins i tot que no entri puntuació

Puntuació incorrecta: El Sistema demana al Comensal una puntuació entre 1 i 5, donant l'opció que no entri puntuació

Plat no en menú: El Sistema demana al Comensal que entri el nom d'un plat del menú, donant l'opció que no entri puntuació



operació valorarPlats(nAmic: String, vpEnt: Int[0..1], vpPri: Int[0..1], vpPos: Int[0..1], d: Date)

pre existeix un Amic *a* amb nom *nAmic*

pre existeix un Menú *m* = (*d*)

pre existeix l'associació *a* menja *m*

pre almenys un de *vpEnt*, *vpPri* o *vpPos* és diferent de nul

pre tots els *vpEnt*, *vpPri*, *vpPos* diferents de nul, tenen un valor entre 1 i 5

pre no existeix cap valoració feta per *a* dels plats inclosos al menú *m*

post si *vpEnt* no és nul, aleshores es crea un instància de Valoració *v* entre *a* i el plat de tipus entrant *pe* inclòs a *m*, amb *v.puntuació* = *vpEnt*

post ...ídem per a plats principals i postres

Modificació de l'apartat 2:

operació platsDisponibles(sNomAmics: Conjunt(String)):
Conjunt(Amic)+Conjunt(String)+Conjunt(String)+Conjunt(String)

pre per a tot *sa* dins *sNomAmics*, existeix en el sistema un Amic amb nom *sa*

pre existeix com a mínim un plat de cada mena (entrant, principal, postre) que no ha estat menjat per cap dels amics de *sNomAmics* en el darrer any **i que no està retirat**

retorna el conjunt d'amics que tenen els noms de *sNomAmics*

retorna el conjunt de noms de plats entrants que no ha estat menjat per cap dels amics de *sNomAmics* en el darrer any **i que no estan retirats**

retorna ídem per plats principals

retorna ídem per plats de postres

operació entraMenú(pEnt: String, vEnt: String, pPri: String, vPri: String, pPos: String, vPos: String, sAmics: Conjunt(Amic), d: Date)

pre *pEnt* és un plat entrant, *pPri* és un plat principal, i *pPos* és un plat de postres

pre *pEnt* combina amb *vEnt*, *pPri* combina amb *vPri*, i *pPos* combina amb *vPos*

pre cap dels amics en *sAmics* ha menjat *pEnt*, ni *pPri*, ni *pPps*, en el darrer any

pre ni *pEnt*, ni *pPri*, ni *pPps*, són plats retirats

post existeix *m*: *Menú* amb data *d* que inclou *pEnt*+*vEnt*, *pPri*+*vPri* i *pPos*+*vPos*

post per tot *a* dins de *sAmics*, *a* menja *m*

Es demana que feu un disseny complet de la solució, aplicant els quatre passos habituals.

Exercici 4. Gestió visites metge

La seguretat social ens ha demanat que li dissenyem una part d'un sistema software per a gestionar les visites que fan els pacients als centres d'assistència primària (CAP), els quals CAPs s'identifiquen per la seva adreça postal. Totes les persones que enregistra aquest sistema són metges o pacients (i un metge pot ser pacient), s'identifiquen pel seu dni i se'n guarda el nom. Aquestes persones

estan assignades com a molt a un CAP (els pacients per visitar-se i els metges per visitar). D'aquesta assignació disposem de la data d'assignació. A més, dels metges se'n sap la seva especialitat, i dels pacients en guardem l'edat. Un pacient programa una visita amb el seu metge en una certa data i hora. Les visites tenen una durada de 1 hora i poden ser de diversos tipus. En el cas de tractar-se d'una visita urgent el sistema enregistra el motiu de la visita. Una visita, un cop feta, no pot variar mai de tipus.

S'obté el següent model de dades, i també alguns casos d'ús (no incloem escenaris alternatius) i model del comportament associat:

cas d'ús Alta Pacient

activació La seguretat social (SS) vol donar d'alta un nou pacient

escenari principal La SS entra el dni, el nom i l'edat d'un nou pacient. El Sistema crea i enregistra el nou pacient

cas d'ús Alta Metge

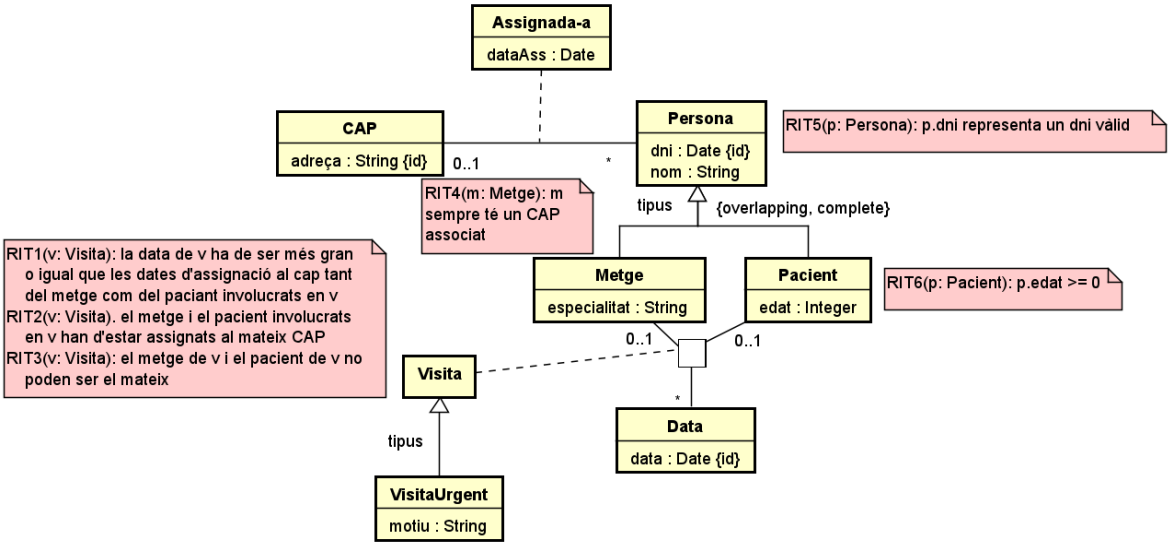
activació La seguretat social (SS) vol donar d'alta un nou metge en un CAP determinat

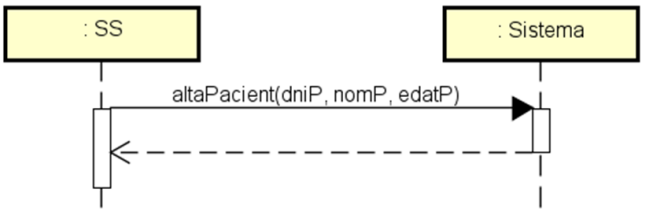
escenari principal El Sistema mostra la llista de CAPs enregistrats (és a dir, la seva adreça postal). La SS entra el dni, el nom i l'especialitat d'un nou metge juntament amb l'adreça postal del CAP d'assignació. El Sistema crea i enregistra el metge i li assigna al CAP seleccionat, enregistrant la data d'avui com a data d'assignació

cas d'ús Programar Visita

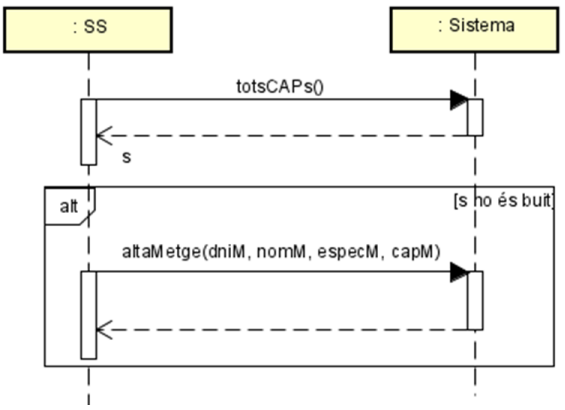
activació Un Pacient vol demanar visita per una especialitat donada

escenari principal El Pacient entra el seu dni i l'especialitat de la que vol visitar-se. El Sistema retorna una llista amb el dni i nom de tots els Metges que tenen disponibilitat per a l'especialitat en el proper mes, ordenada per proximitat de data. El Pacient entra el dni d'un dels metges i el Sistema programa la visita per la primera data disponible del Metge



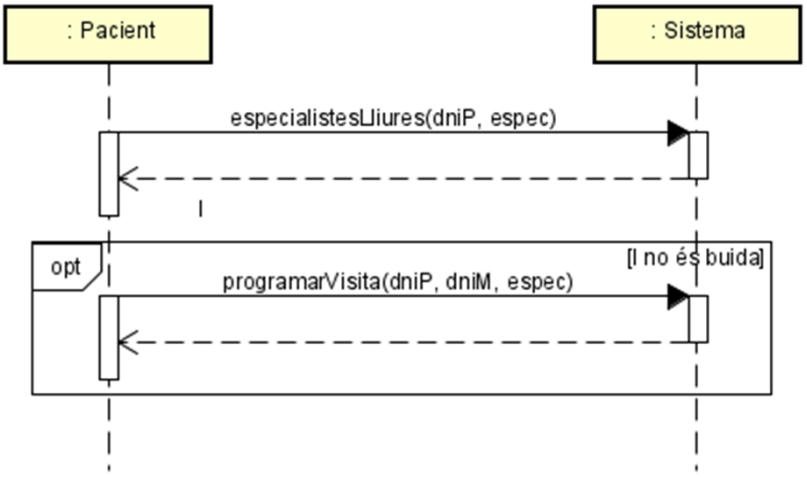


operació altaPacient(dniP: String; nomP: String; edatP: Integer)
post existeix un Pacient p = (dniP, nomP, edatP)



operació totCAPs(): Set(String)
retorna el conjunt d'adreces postals de tots els CAP existents en el sistema

operació altaMetge(dniM: String; nomM: String; especM: String; adC: String)
pre adC és l'adreça postal d'algun CAP donat d'alta al sistema
post existeix un Metge m = (dniM, nomM, especM)
post existeix una associació (m, c, d) tal que c.adreça = adC i d = avui()



operació especialistesLliures(dniP: String; espec:String): List(String+String)
pre existeix un Pacient amb dni = dniP
retorna la llista de dnis i noms dels metges que són d'especialitat espec, que són al mateix CAP que dniP, i que tenen disponibilitat per visitar durant el proper mes, ordenats per data de primera visita disponible (aleatori, en cas d'empats)

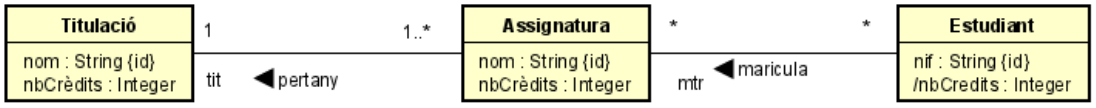
operació programaVisita(dniP: String; dniM: String, espec: String)
pre existeix un Pacient amb dni = dniP
pre existeix un Metge amb dni = dniM de l'especialitat espec i el mateix CAP que dniP que té visita lliure el mes vinent

post existeix una visita v = (m, p, d) on m.dni = dniM, p.dni = dniP, d.data = primera data disponible del metge m, d.h = primera hora disponible del metge m en la data d.data

Es demana que feu un disseny complet de la solució, aplicant els quatre passos habituals. Useu controladors cas d'ús a les capes de presentació i domini.

Exercici 5. Matriculació docent

Una escola universitària vol gestionar les assignatures que formen les seves titulacions i de les què es matriculen els seus estudiants segons el model de dades següent:



ID (e: Estudiant): e.nbCrèdits = suma dels.nbCrèdits de totes les assignatures en les què e està matriculat.

Considerem els casos d'ús següents (concrets, i.e. decorats amb elements de presentació), que generen un model de comportament de l'especificació:

cas d'ús Nova Matrícula

activació L'Estudiant vol matricular-se d'una assignatura

escenari principal

1. L'Usuari entra el seu nif i el nom de l'assignatura de la què es vol matricular en sengles camps de dades i prem OK
2. El Sistema enregistra la matrícula

cas d'ús Graduats

activació El Cap d'Estudis vol saber els estudiants graduats d'una titulació

escenari principal

1. El Cap d'Estudis entra el nom d'una titulació TIT en un camp de dades i prem OK
2. El Sistema mostra per pantalla el conjunt dels nifs dels estudiants graduats en la titulació TIT (és a dir, que el nombre de crèdits matriculats per l'estudiant en assignatures de TIT és més gran o igual que el nombre de crèdits de TIT)

cas d'ús Prou Oferta?

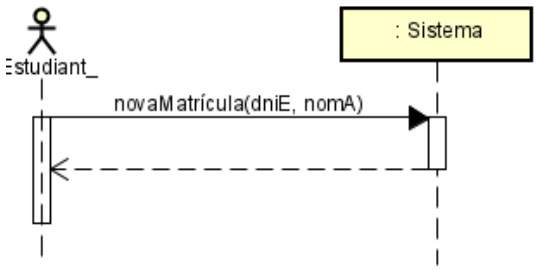
activació El Cap d'Estudis vol saber si hi ha prou assignatures ofertades en una titulació com per tenir-ne estudiants graduats

escenari principal

1. El Cap d'Estudis entra el nom d'una titulació TIT en un camp de dades i prem OK
2. El Sistema mostra per pantalla "Sí" si la suma del nombre de crèdits de les assignatures de TIT és més gran o igual que el nombre de crèdits de TIT, i "NO" altrament

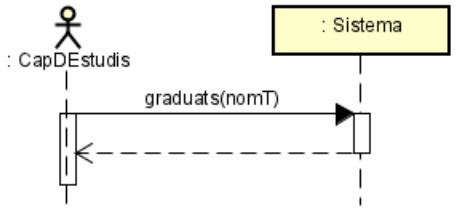
A continuació trobeu el model del comportament pels tres casos d'ús anteriors.

Cas d'ús Nova Matrícula



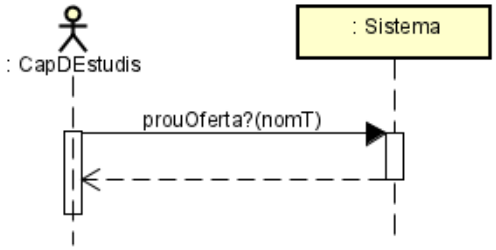
operació novaMatrícula
(dniE: String, nomA: String)
pre existeix un Estudiant e amb e.dni = dniE
pre existeix una Assignatura a
amb a.nom = nomA
post es crea una matrícula m = (e, a)

Cas d'ús Graduats



operació graduats(nomT: String): Set(String)
pre existeix una Titulació t amb t.nom = nomT
retorna els e.nif de tots els Estudiants e tals que el número de crèdits matriculats en assignatures de t sigui més gran o igual a t.nbCrèdits

Cas d'ús Prou Oferta?



operació prouOferta?(nomT: String): Bool
pre existeix una Titulació t amb t.nom = nomT
retorna cert si la suma de a.nbCrèdits de les totes Assignatures a que pertanyen a t, és més gran o igual que t.nbCrèdits, altrament retorna fals

Es demana que feu el disseny d'aquest sistema, incloent-ne: (a) el diagrama de classes de disseny, (b) els diagrames de seqüència corresponents als casos d'ús i (c) els contractes de les operacions que surtin en aquests diagrames. Useu controlador façana (singleton) a la capa de presentació i transacció a la capa de domini .

Exercici 6. L'acadèmia informàtica

L'acadèmia Bits4All vol obrir una nova línia de cursos centrats en intel·ligència artificial (IA), per exemple sobre *Xarxes Neuronals* o *ChatGPT*. De cada curs, Bits4All va obrint noves edicions quan ho considera convenient. Els estudiants es poden pre-inscriure de diversos cursos, i si compleixen certs requisits (detallats més endavant), poden matricular-se en una edició del curs.

Requisits de dades:

- Cursos. Dels cursos se'n sap el seu nom, el codi (que els identifica), número d'hores, i preu. El preu es pot calcular com el producte del número d'hores del curs per un preu per hora únic que determina l'acadèmia.
- Edicions de cursos. Cada edició té una data d'inici, una data de fi, una seu on s'imparteix, i una capacitat (número d'estudiants que s'hi poden matricular). Dues edicions d'un mateix curs no es poden solapar en el temps.
- Estudiants. Dels estudiants se'n sap el seu dni (que els identifica), nom i expertesa: titulació universitària (cap, enginyeria, màster, doctorat; si en té més d'una, ens quedem amb la superior) i coneixement en almenys una àrea informàtica (p.e., Enginyeria del Software, Ciència de Dades, Aprenentatge Automàtic, entre d'altres; la llista no és tancada).

Restriccions:

- L'oferta de cursos es crea en posar en marxa el sistema i no pot canviar.
- No existeix cap funcionalitat explícita de donar d'alta un estudiant.
- Els estudiants no renuncien mai a una pre-inscripció ni a una matrícula.
- El procés de determinar si una pre-inscripció compleix els requisits demanats pot durar uns quants dies.
- El preu per hora determinat per l'acadèmia no canviarà al llarg de la vida del sistema.
- Les àrees d'expertesa de l'estudiant no poden canviar.

Bits4All vol assegurar-se que els estudiants tenen els coneixements necessaris per seguir els cursos. A tal efecte, se sap de cada curs quines àrees informàtiques cobreix (p.e., el curs *Xarxes Neuronals* cobreix l'àrea d'*Aprenentatge Automàtic*). Si un estudiant es pre-inscriu en un curs sense tenir-ne els coneixements, la pre-inscripció és rebutjada i es guarda la data de la decisió de rebuig.

Si una preinscripció s'accepta, l'estudiant és matrícula a la edició del curs més propera en el temps, que no ha arribat a omplir la seva capacitat. A més, se li assigna un tutor acadèmic (només cal guardar el nom del tutor).

Un cop acaba l'edició del curs, l'acadèmia enregistra la nota de tots els seus estudiants (entre 0 i 10). Un estudiant que no aprova (nota < 5) té el dret de tornar a matricular-se en una nova edició del curs. L'estudiant no té més que aquestes dues oportunitats per aprovar el curs.

Si un estudiant té aprovat tots els cursos oferts per Bits4All que cobreixen una àrea determinada, es considera usuari premium i l'acadèmia li ofereix un descompte (percentatge sobre el preu del curs) en matrícules posteriors. El valor d'aquest descompte no està regulat, i l'acadèmia pot oferir descomptes diferents a estudiants diferents. A més, Bits4All ofereix a alguns d'aquests estudiants premium la possibilitat d'actuar com a consultors en aquestes àrees on tenen tots els cursos aprovats.

Bits4All vol un sistema que inclou, entre d'altres, les funcionalitats següents:

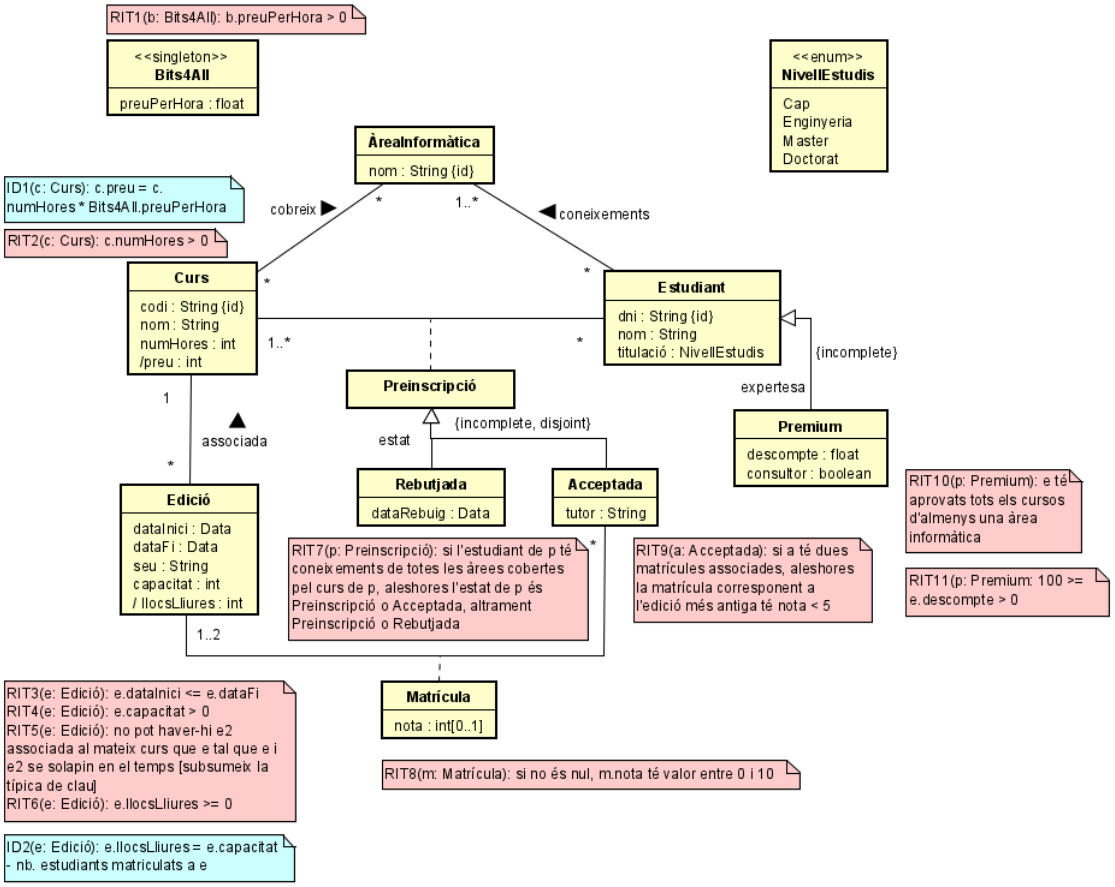
CU1. Pre-inscriure un estudiant en un curs ja existent. L'estudiant gestiona la seva pre-inscripció usant la página web de l'acadèmia. Si no estava registrat en el sistema, ha de proporcionar les seves dades en aquest cas d'ús.

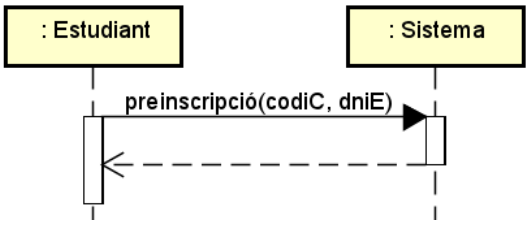
CU2. Acceptar la pre-inscripció. Bits4All accepta la pre-inscripció que ha fet un estudiant amb anterioritat.

CU3. Cursos plens. Bits4All vol saber quins cursos tenen exhaurida la capacitat de totes les seves edicions obertes que estan per celebrar-se.

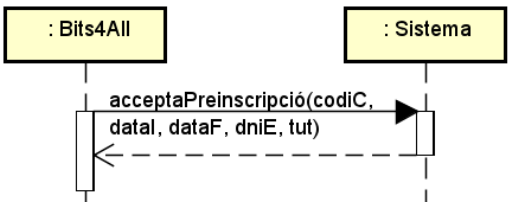
CU4. Estudiants premium. Bits4All vol saber quins estudiants Premium estan matriculats en algun curs encara no aprovat.

Considereu el següent model d'especificació, considereu que a tots els casos d'ús l'usuari entra la informació en camps de text (sense llistes pre-carregades amb valors possibles):

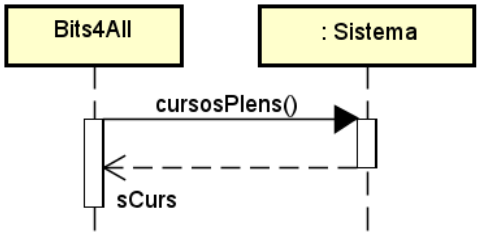




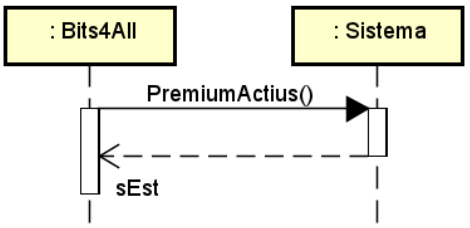

```
operació preinscripció
         (codiC: String; dniE: String)
pre existeix Curs c amb c.codi = codiC
pre existeix Estudiant e amb e.dni = dniE
post enregistra que e es preinscriu de c
```




```
operació acceptar(codiC: String;
           dataI, dataF: Data;
           dniE: String; tut: String)
pre l'estudiant e amb e.dni = dniE està
      preinscrit del curs c amb c.codi =
      codC amb preinscripció p ni rebutjada
      ni acceptada
pre existeix una edició d del curs c amb
      d.dataInici = dataI i d.dataFi = dataF
post especialitza p com a acceptada
post p.tutor = tut
post crea una matrícula entre p i d
```




```
operació cursosPlens(): Conjunt(String)
retorna els noms dels cursos c tal que totes
      les seves edicions per celebrar estan
      plenes
```



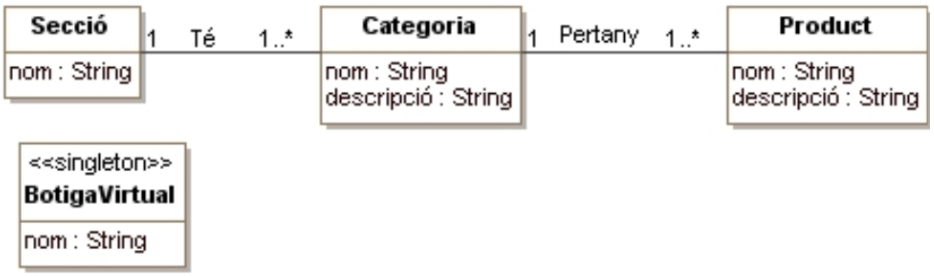

```
operació premiumActius(): Conjunt(String)
 retorna els noms dels estudiants Premium que
      estan matriculats en algun curs encara
      no aprovat
```

Es demana que feu el disseny d'aquest sistema. Useu controladors façana (singleton) a la capa de presentació i transacció a la capa de domini.

PART II. EXERCICIS DE LA CAPA DE DOMINI

Exercici 7. La botiga virtual

Volem dissenyar una botiga virtual. Aquesta botiga ven productes que tenen un nom i una descripció. Els productes pertanyen a categories que també tenen el seu nom i la seva descripció. Les categories pertanyen a seccions de la botiga. A continuació disposeu de l'esquema conceptual del sistema:



Les restriccions de clau són: (Secció, nom); (Categoria, nom); (Producte, nom).

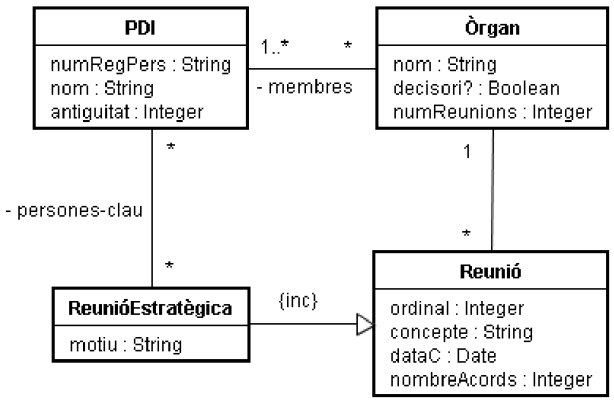
La capa de domini ofereix la següent operació:

```
context CapaDeDomini::obtenirInformacióSecció (nomS: String):
          Conjunt((String + String + Conjunt(String+String)))
exc seccióNoExisteix: la secció identificada per nomS no existeix
retorna el nom i la descripció de les categories associades a la secció nomS i
   per a cada categoria, el nom i descripció dels productes de la categoria.
```

Feu el disseny de la capa de domini usant controladors transacció.

Exercici 8. Reunions a la UPC

Es vol tractar el problema de la gestió de les reunions de la UPC. El diagrama de classes següent mostra que els Òrgans (identificats per un string, e.g. Consell de Departament, Junta d'Escola, …) poden ser decisoris i enregistren el nombre de reunions fetes, que les Reunions enregistren el nombre d'acords que s'hi van prendre (atribut derivat ja materialitzat), i que s'enregistren els PDI (és a dir, els professors, identificats per el seu número de registre personal) que són persones clau en un tipus particular de reunions, les Reunions Estratègiques (que són estratègiques per un motiu determinat). Notem que la clau de les reunions és multi-atribut i dèbil:



Restriccions d'integritat de clau (la resta no són importants per al problema):

PDI □ numRegPers
Òrgan □ nom
Reunió □ Òrgan::nom+
ordinal+concepte

Ens interessen les dues operacions següents:

operació CapaDeDomini::novaReunióEstratègica(nomO: String; concR: String; ordR: Int; dataR: Date; nacR: Int; motR: String; keyP: Conjunt(String))

pre PDIs-existeixen: tots els números de registre personal del conjunt keyP són de PDI existents al sistema

exc òrgan-no-existeix: l'òrgan nomO no existeix

exc reunió-existeix: la reunió nomO+concR+ordR existeix

post es dóna d'alta una reunió estratègica per a l'òrgan nomO amb concepte concR i ordinal ordR, motiu motR, data dataR, nombre d'acords nacR i persones clau keyP

post s'incrementa el nombre de reunions enregistrat per a l'òrgan nomO

operació CapaDeDomini::llistat(nrpP: String): Conjunt(String + Conjunt(String+Int))

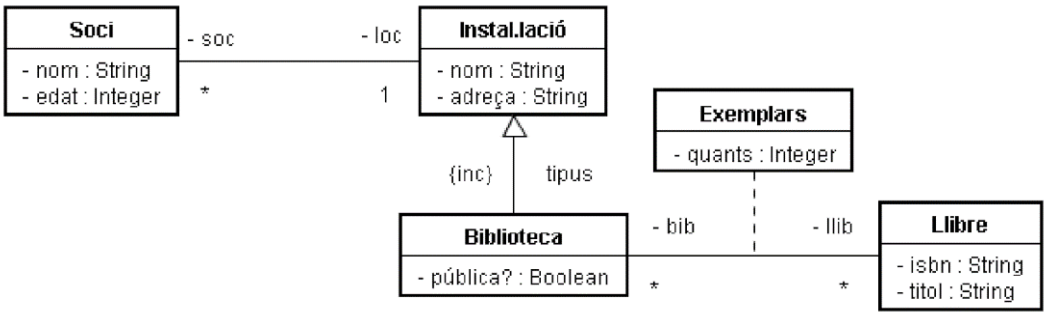
pre PDI-existeix: existeix el PDI nrpP

post per a cada òrgan decisori de la UPC retorna el seu nom amb el conjunt de reunions estratègiques (concepte i ordinal) en les què nrpP era persona clau i no s'hi ha pres cap acord

Feu el disseny de la capa de domini usant controladors transacció.

Exercici 9. La biblioteca

Apartat 1. Es vol implementar el tros de model conceptual següent:



Les claus són: (Soci, nom); (Instal.lació, nom); (Llibre, isbn).

Ens interessen les dues operacions següents:

```
operació CapaDeDomini::compraLlibres(isbnLl: String, nomB: String, q: Integer)
exc llibre-no-existeix: el llibre isbnLl no existeix
exc biblioteca-no-existeix: la Biblioteca nomB no existeix
exc cap-exemplar: q <= 0
post enregistra l'adquisició de q exemplars del Llibre isbnLl per la Biblioteca
     nomB; noteu que la Biblioteca ja podia (o no) tenir algun exemplar del
     Llibre
```

```
operació CapaDeDomini::bibliotecaPúblicaAmbMésSocis(): String
exc sense-biblioteques-públiques: no hi ha cap Biblioteca pública en el
                                  sistema
retorna el nom de la Biblioteca pública amb més Socis (en cas d'empat,
        retorna qualsevol de les empatades)
```

Feu el disseny de la capa de domini usant controladors transacció.

Apartat 2

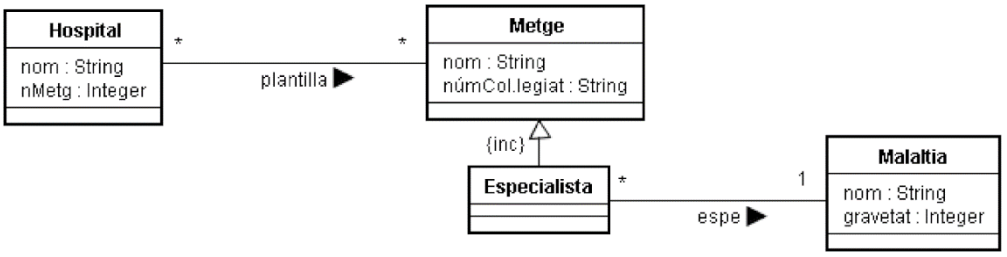
Repetiu l'apartat 1 per les dues operacions següents:

```
operació CapaDeDomini::consultaExemplars(isbnLl: String, nomB: String): Integer
exc llibre-no-existeix: el Llibre isbnLl no existeix
exc biblioteca-no-existeix: la Biblioteca nomB no existeix
exc no-hi-ha-exemplars: no hi ha cap exemplar del llibre a la biblioteca
retorna el nombre d'exemplars que hi ha del llibre isbnLl a la Biblioteca nomB
```

```
operació CapaDeDomini::novaBiblioteca(nomB: String, adrc: String, pub: Bool)
exc biblioteca-ja-existeix: la Biblioteca nomB ja existeix
exc instal.lacio-ja-existeix: nomB correspon a un codi d'instal·lació existent
post enregistra la Biblioteca nomB a la base de dades
```

Exercici 10. Malalties

Es vol implementar el tros de model conceptual següent:



Les restriccions de clau són: (Hospital, nom); (Metge, nom); (Malaltia, nom).

Ens interessen les dues operacions següents:

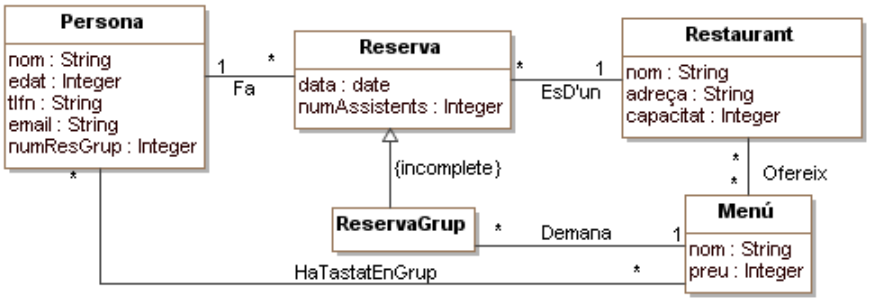
```
operació CapaDeDomini::metgePlega(nomH: String, nomM: String)
pre el Metge nomM existeix i no és Especialista
pre l'Hospital nomH existeix
exc metge-no-hi-treballa: el Metge nomM no treballa a l'Hospital nomH
post enregistra que el Metge nomM deixa de treballar a l'Hospital nomH
post si el Metge nomM només és a la plantilla de l'hospital nomH, l'esborra
post decrementa el nombre de metges en plantilla de l'hospital, nMetg
```

```
operació CapaDeDomini::malaltiaMésGreu(): String
exc no-hi-ha-malalties: no hi ha cap Malaltia enregistrada en el sistema
retorna el nom de la Malaltia que té gravetat més gran
```

Feu el disseny de la capa de domini usant controladors transacció.

Exercici 11. La cadena de restaurants

Una cadena de restaurants que ofereix menús per sopars ens ha demanat que li dissenyem una part d'un sistema software per gestionar les seves reserves. L'esquema conceptual (de l'especificació) es mostra a continuació:



Restriccions d'integritat
RI1 - Claus: (Persona, nom); (Restaurant, nom); (Menú, nom); (Reserva, Persona::nom+data);
RI2 - Es considera que no hi pot haver dos persones amb el mateix email.
RI3 - Els menús de les reserves de grup han de ser menús oferts pel restaurant on s'ha fet la reserva.
RI4 - La capacitat d'un restaurant ha de ser més gran que el sumatori dels assistents de les reserves previstes per aquell restaurant en una data determinada.
RI5 - El numAssistents d'una reserva de grup ha de ser més gran que 5.
RI6 - Els menús que ha tastat en grup una persona són els mateixos que ha reservat.
RI7 - El numResGrup d'una persona és igual al número de reserves de grup que ha fet la persona.
RI8 - En numResGrup ha de ser més gran o igual a 0. La resta d'atributs de tipus integer han de ser positius.
RI9 - Els únic atributs que apareixen en el diagrama de classes pels que es permet valors nuls és l'atribut edat, i per un dels dos atributs tlfn, email però mai per tots dos.

La capa de domini ofereix l' operació següent:

```
operació novaReservaGrup(nomClient: String, nomRest: String, dr: Date,
                                          nAss: Integer, nomMenú: String): Integer
pre client-existeix, data-ok, més-5assistents, reserva-no-existeix
exc restaurant-no-existeix: el restaurant amb nomRest no existeix.
exc menu-no-ofert: el restaurant nomRest no ofereix el menú nomMenú.
exc restaurant-sense-lloc: el restaurant nomRest no té disponibilitat pel nAss a
     la data dr.
post reserva-grup-creada: es crea una reserva de grup i es formen totes les
     associacions amb la persona, el restaurant i el menú.
post numResGrup-incrementat: s'incrementa el numResGrup de la persona nomClient.
post haTastat-creat: si la persona no ha tastat el menú, es crea una instància de
     HaTastatEnGrup entre la persona nomClient i el menú nomMenú.
post result= número de places disponibles del restaurant nomRest a la data dr
     (capacitat del restaurant - número de assistents (de les reserves) per la
     data dr - nAss).
```

Feu el disseny de la capa de domini usant controladors transacció.