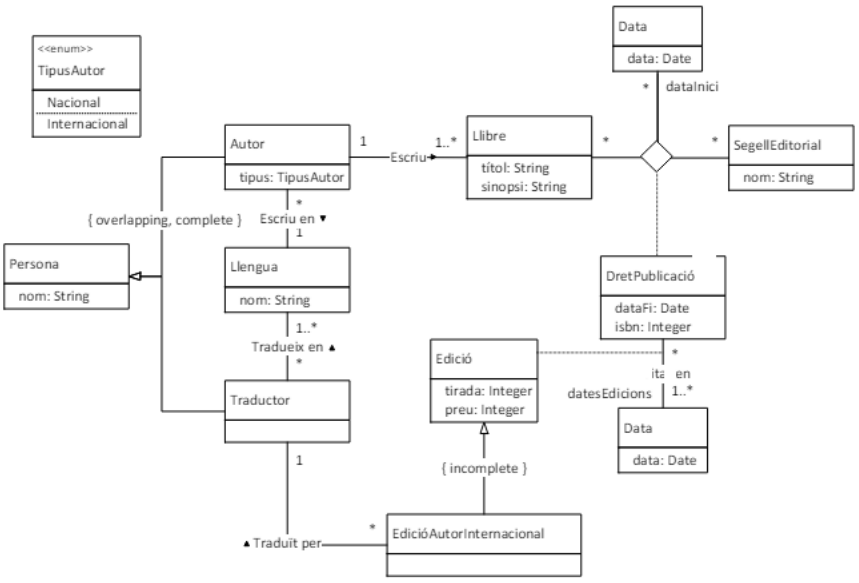
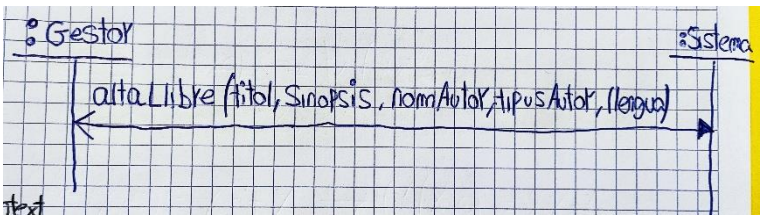


Restriccions Textuals

1. Claus externes: (Persona, nom), (Llibre, títol), (SegellEditorial, nom), (Data, data), (Llengua, nom).
2. Si un traductor és també autor, la llengua en què escriu ha de ser una de les que tradueix.
3. Un segell editorial pot adquirir els drets d'un llibre en diferents períodes però no sobreposats en el temps.
4. La data d'inici d'un dret de publicació ha de ser anterior a la seva data de fi.
5. L'isbn no es pot repetir entre tots els drets de publicació.
6. Les dates de les edicions d'un llibre han d'estar compreses entre les seves dates de dret de publicació.
7. El traductor d'una edició d'autor internacional ha de traduir la llengua en què escriu l'autor del llibre.
8. Una edició és d'autor internacional si i només si correspon a un llibre que té un autor de tipus internacional.

El sistema a desenvolupar no ha de donar d'alta les dades de Traductor, Data, Llengua i SegellEditorial, ja que existeix un altre sistema que ho fa. El sistema ha de permetre efectuar les funcionalitats següents:

Alta de Llibre: Quan l'editor vol enregistrar les dades d'un nou llibre i el seu autor, li comunica a l'empleat gestor de continguts que introdueixi les dades necessàries per a l'alta del llibre. Cal destacar que si l'autor no existia, es crea en aquest mateix moment, i que si era traductor però encara no era autor, caldrà introduir en quina llengua escriu (que, lògicament, ha de ser una de les que tradueix).



Context: Sistema::**AltaLlibre** (***títol***: String, ***sinopsi***: String, ***nomAutor***: String, ***tipusAutor***: TipusAutor, ***llengua***: String)

Pre: **Llengua**.allInstances()->**exists**(l | **l.nom = llengua**)

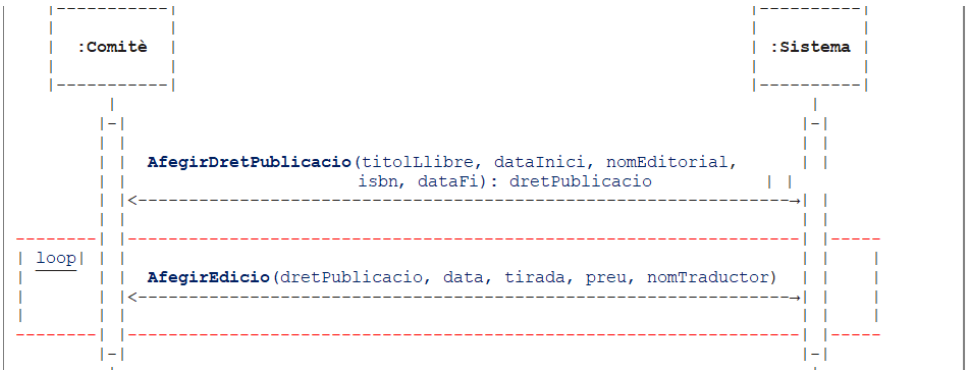
Post:

- if not **Persona**.allInstances()**@pre**->**exists**(p | **p.nom = nomAutor**) then **Autor**.allInstances()->**exists**(a | **a.oclIsNew()**
  and **a.nom = nomAutor**
  and **a.tipus = tipusAutor**
  and **a.llengua.nom = llengua**

  endif
- and if **Persona**.allInstances()**@pre**->**exists**(p | **p.nom = p.nomAutor** and
  p.oclIsTypeOf(Traductor) and
  not **p.oclIsTypeOf(Autor)** and
  p. oclAsTypeOf(Traductor).llengua.nom -> **includes(llengua)**)
  then **Autor**.allInstances()->**exists**(a | **a.nom = a.nomAutor**
  and **a.tipus = tipusAutor**
  and **a.llengua.nom** = **llengua**)

  endif
- and **Llibre**.allInstances()->**exists**(l | **l.oclIsNew()** and **l.titol = titol** and **l.sinopsi = sinopsi** and **l.autor.nom = nomAutor**)

Adquisició de Dret de Publicació: Quan el comitè de publicació ha adquirit els drets de publicació d'un llibre introdueix les dades necessàries i hi entra també les dades de les diferents edicions. Feu que la interacció necessària per dur a terme aquesta funcionalitat requereixi més d'un esdeveniment.



Context: Sistema::**AfegirDretPublicacio** (**titolLlibre**: String, **dataInici**: Date, **nomEditorial**: String, **isbn**: int, **dataFi**: Date) : ***:DataPublicacio***

Pre:

Data.allInstances()->**exists**(d | **d.data = dataInici**)

SegellEditorial.allInstances()->**exists**(s | **s.nom = nomEditorial**)

Llibre.allInstances()->exists(l | **l.titol = titolLlibre**)

Post:

DretPublicacio.allInstances()->**exists**(p | **p.oclIsNew()**

and **p.LLibre.titol = titolLlibre**

and **p.data.data = dataInici**

and **p.segellEditorial.nom = nomEditorial**

and **p.dataFi = dataFi**

and **p.isbn = isbn**

and **return = p**)

Context: Sistema::**AfegirEdicio** (**dretPublicacio**: DretPublicacio, **data**: Date, **tirada**: int, **preu**: int, **nomTraductor**: String)

Pre:

Data.allInstances()->**exists**(d | **d.data = data**)

Post:

Edicio.allInstances()->**exists**(e | **e.oclIsNew()**

and **e.tirada = tirada**

and **e.preu = preu**

and **e.data.data = data**

and **e.dretPublicacio = dretPublicacio**

and if (**e.dretPublicacio.llibre.autor.tipus = TipusAutor::Internacional**) then

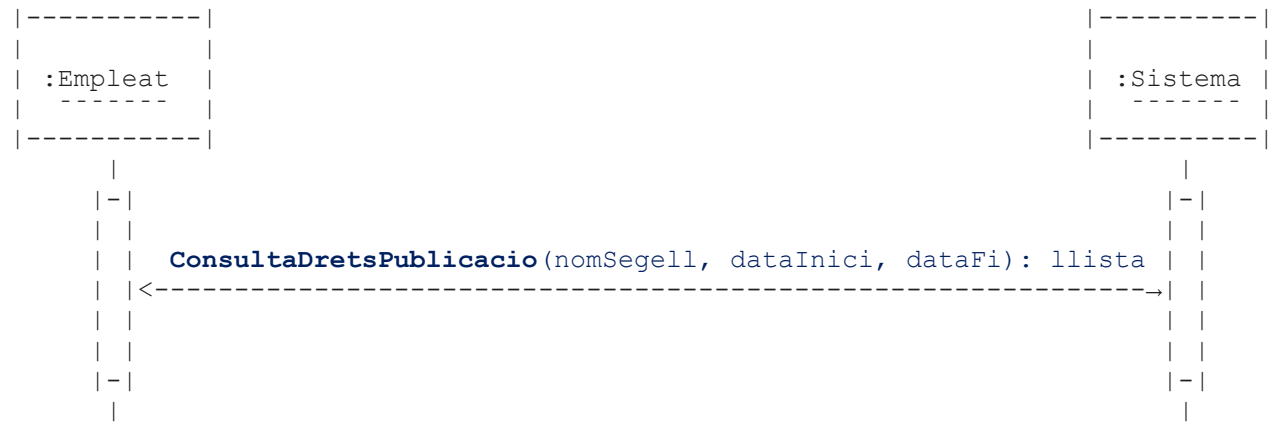
e.oclIsTypeOf(EdicioAutorInternacional)

and **e.oclAsType(EdicioAutorInternacional).traductor.nom = nomTraductor**

endif

)

Consulta Drets de Publicació: Quan l'editor vol obtenir el llistat de drets de publicació d'un segell editorial en un període de temps, ell mateix indica el nom del segell i les dates d'inici i fi del període al sistema. El llistat mostra, per cada dret de publicació adquirit pel segell editorial i tal que la seva data d'inici està dins del període i el nombre d'edicions és major que 5, la següent informació: el títol del llibre, l'isbn, i la llista de dates de totes les seves edicions. Aquesta funcionalitat només es pot demanar si hi ha com a mínim 10 llibres diferents amb drets de publicació pel segell durant el període.



Context: Sistema::**ConsultaDretsPublicacio** (**nomSegell**: String, **dataInici**: Date, **dataFi**: Date): ***Set*** (***TupleType*** (**titolLlibre**: String,

isbn: String,

datesEdicions: Set(Date)))

Pre:

Data.allInstances()->**exists**(d | **d.data = dataInici**)

Data.allInstances()->**exists**(d | **d.data = dataFi**)

DretPublicacio.allInstances()->**exists**(d | **d.segellEditorial.nom = nomSegell**

and **d.Data.data >= dataInici**

and **d.Data.data < dataFi**

and **d.llibre.asSet()->size() >= 10**)

Body:

let llista: **Set(DretPublicacio)** = **DretPublicacio**.allInstances()->**select**(p | **p.dataInici >= dataInici**

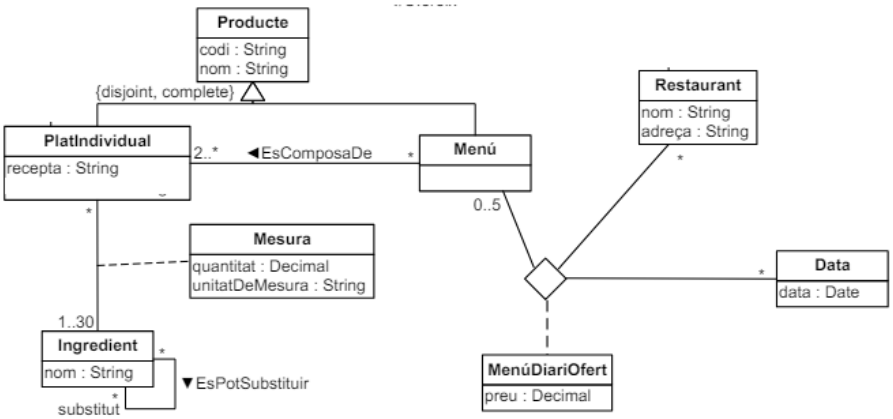
and **p.dataFi < DataFi**

and **p.segellEditorial = nomSegell**

and **p.datesEdicions->size() > 5**)

in

llista->**collect**(p |***Tuple*** { **titolLlibre = p.llibre.nom**, **isbn = p.isbn**, **datesEdicions = p.datesEdicions** })

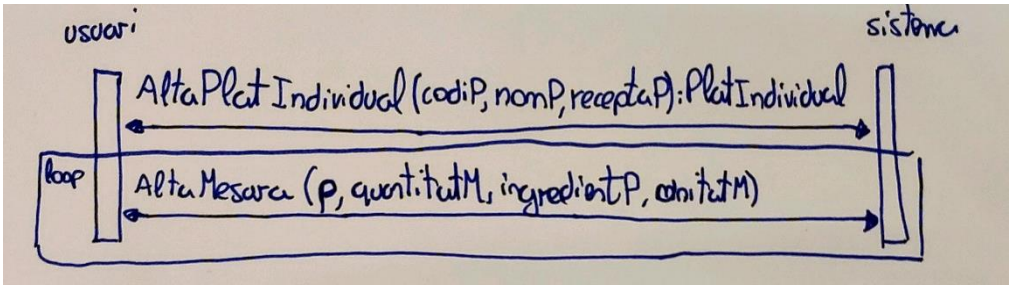


Restriccions d'integritat textuals:

1. Claus externes: (Producte, nom), (Ingredient, nom), (Restaurant, nom), (Data, data)
2. Un restaurant ofereix com a màxim 50 menús diaris diferents.
3. Un ingredient no pot ser un substitut d'ell mateix

El sistema a desenvolupar no ha de donar d'alta restaurants, ingredients ni dates atès que hi ha un altre sistema encarregat de fer-ho. En canvi, si que ha de proporcionar les funcionalitats següents: AltaPlatIndividual, ConfecciónMenúDiari i ConsultaPlatsOferts

Quan un usuari vol donar d'alta un plat individual, indica al sistema tota la informació necessària per a fer-ho. És a dir, tota la informació del plat individual i de tots els ingredients que en formen part. Feu que la interacció necessària per a portar a terme aquesta funcionalitat requereixi més d'un esdeveniment.



CONTEXT: Sistema::**AltaPlatIndividual**(**codiP**:String, **nomP**:String, **receptaP**:String, **ingredientsP**:Set(Strings)) :***PlatIndividual***

PRE: **Ingredient**.AllInstances().**nom** -> **includesAll**(**ingredientsP**)

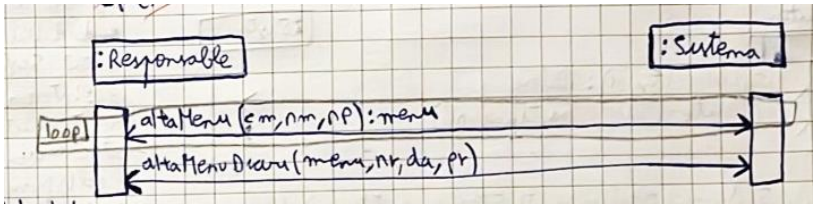
POST: **PlatIndividual**.AllInstances()-> **exists**(p | **p.oclIsNew()**
and **p.nom = nomP**
and **p.codi = codiP**
and **p.receptaP = receptaP**
and **p.Ingredient = ingredientsP**
and **result = p**)

CONTEXT: Sistema::**AltaMesura**(**p**:PlatIndividual, **quantitatM**:Decimal, **ingredientP**:String, **unitatM**:String)

PRE: **Ingredient**.AllInstances() -> **exists**(i | **i.nom = ingredientP**)

POST: **Mesura**.AllInstances() -> **exists**(m | **m.oclIsNew()**
and **m.quantitat = quantitatM**
and **m.unitatDeMesura = unitatM**
and **m.Ingredient.nom = ingredientP**
and **m.PlatIndividual = p**)

Quan el responsable de cuina vol donar d'alta un menú diari ofert, indica al sistema tota la informació necessària per a fer-ho. A més, si el menú del menú diari ofert no existeix aleshores també s'haurà de donar d'alta i s'haurà d'associar als plats individuals de què es composa (que suposarem que han d'existir prèviament a l'alta del menú). Cal tenir en compte que un restaurant no pot oferir més de 50 menús diaris diferents i que no es pot donar d'alta un nou menú diari si el restaurant ja ofereix més de 100 plats individuals.



CONTEXT: Sistema:: **AltaMenu**(**cm**: String, **nm**: String, **np**: String): **Menu**

PRE: **PlatIndividual**.allInstances() ->**exists**(p|**p.nom = np**)

POST: if not **Menu**. allInstances**()@pre** -> **exists** (m|**m.nom = nm**) then

Menu. allInstances() -> **exists** (m1|**m1.nom = nm** and **m1.codi = cm** and **m1.oclIsNew**())

Endif

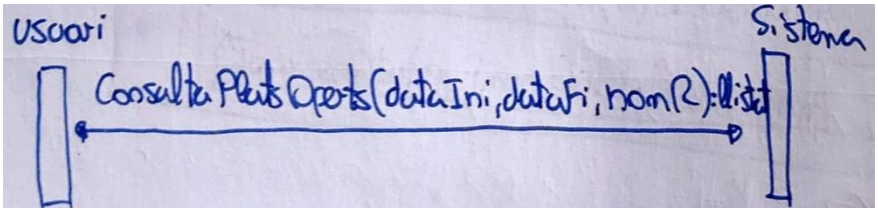
Menu.allInstances() -> **exists** (m| **m.nom = nm** and **m.PlatIndividual.nom** ->**includes(np)** and **result = m**)

CONTEXT: Sistema::**AltaMenuDiari**(**menu**:Menu, **nomP**:Decimal, **dataM**:Date, **nomR**:String)

PRE: **Restaurant**.AllInstances() -> **exists** (r | **r.nom = nomR** and
r.MenuDiariOfert ->size() < 50
and **r.Menu.PlatIndividual -> asSet() -> size() <= 100**)

POST: **MenúDiariOfert**.AllInstances() -> **exists** (md | **md.oclisNew()**
and **md.Data.data = dataM**
and **md.Restaurant.nom = nomR**
and **md.Menu = nomP**
and **md.preu = preuM)**

Quan un usuari vol fer una consulta dels plats de la temporada, indica una data d'inici, una data fi i el nom del restaurant per al qual vol fer la consulta. El sistema li retorna un llistat que conté per cada plat individual ofert a algun menú del restaurant durant el període entre la data d'inici i data de fi, el seu codi i el nom dels seus ingredients. Aquest llistat només s'emetrà si el restaurant ofereix com a mínim deu menús diaris d'un preu inferior a 10€.



CONTEXT: Sistema::**ConsultaPlatsOferts**(**dataIni**:Date, **dataFi**:Date, **nomR**:String)
:**Set(TupleType**(**codiPr**:String, **ingredientsPr**:Set(String)))

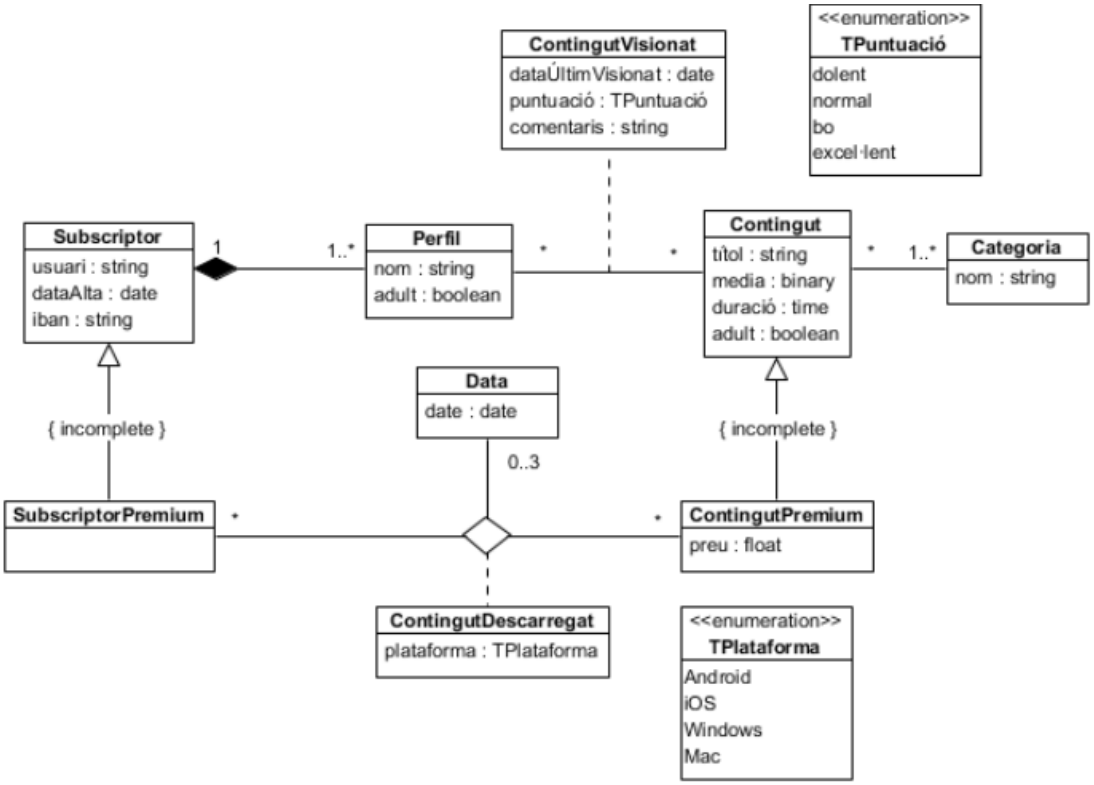
PRE: **dataIni < dataFi** and
Restaurant.AllInstances() -> **exists**(r | **r.nom = nomR** and **r.MenúDiariOfert** -> **select**(md | **md.preu < 10**) -> **size() >= 10)**

BODY: let llistat: **Set(PlatIndividual)** = **PlatIndividual**.AllInstances() -> **select**(p | **p.Menu.Restaurante.nom = nomR**
and **p.Menu.Data.date > dataIni**
and **p.Menu.Data.date < dataFi**)

in

result = llistat -> **collect**(ll | Tuple { **codiPr = ll.codi,**

ingredients = ll.Ingredient })

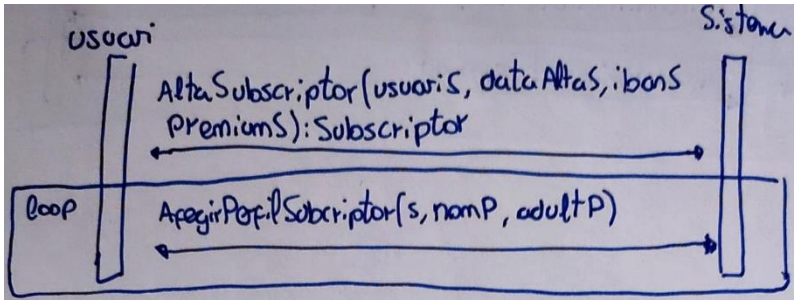


Restriccions Textuals

1. Claus externes: (Subscriptor, usuari), (Contingut, títol), (Categoria, nom), (Data, data)
2. Un subscriptor no pot tenir dos perfils amb el mateix nom.
3. Un perfil no adult no pot visionar contingut adult
4. La data de visionat d'un contingut ha de ser posterior a la data d'alta del subscriptor del perfil de visionat
5. La data de descàrrega d'un contingut ha de ser posterior a la data d'alta del subscriptor

El sistema a desenvolupar no pot modificar les dades de Categoria, Contingut, ContingutPremium, i Data, ja que existeix un altre sistema que les gestiona. El sistema ha de permetre efectuar les següents funcionalitats:

Alta de Subscriptor: Quan un usuari vol inscriure's a la plataforma ell mateix introdueix el seu nom d'usuari i el seu IBAN. A més, indica si vol accedir al contingut premium. Una vegada donat d'alta, l'usuari crea diferents perfils, tot indicant-ne el seu nom i si podrà accedir a contingut adult. Feu que la interacció necessària per dur a terme aquesta funcionalitat requereixi dos esdeveniments.



CONTEXT: Sistema::**AltaSubscriptor**(**usuariS**:String, **dataAltaS**:String, **ibanS**:String, **premiumS**:Bool) :***Subscriptor***

PRE:

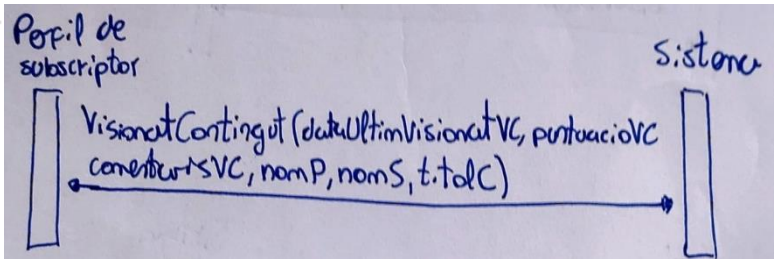
POST: **Subscriptor**.AllInstances()-> **exists**(s | **s.oclIsNew()**
and **s.usuari = usuariS**
and **s.dataAlta = dataAltaS**
and **s.iban = ibanS**
and **if premium = true** then **s.oclIsTypeOf(SubscriptorPremium)**
endif
and **result = s**)

CONTEXT: Sistema::**AfegirPerfilsSubscriptor**(**s:** Subscriptor, **nomP**:String, **adultP**:Bool)

PRE:

POST: **Perfil**.AllInstances()-> **exists**(pr | **pr.oclIsNew()** and **pr.nom = nomP** and **p.adult = adultP** and **pr.Subscriptor = s**)

Visionat de Contingut: Quan un perfil de subscriptor realitza un visionat d'un contingut, introdueix al sistema les dades necessàries per fer-ho. Si el perfil de subscriptor ja havia visionat aquest contingut anteriorment, aleshores les dades d'aquest últim visionat sobreescriuen les anteriors. Altrament, es crea una instància de la classe ContingutVisionat amb les dades corresponents. Aquesta funcionalitat no es pot portar a terme si la data de visionat és anterior a la data d'alta del subscriptor del seu perfil. Feu que la interacció necessària per dur a terme aquesta funcionalitat requereixi un únic esdeveniment.



CONTEXT: Sistema::**VisionatContingut**(**dataUltimVisionatVC**:Date, **puntuacioVC**:TPuntuacio, **comentarisVC**:String, **nomP**:String, **nomS**:String, **titolVC**:String)

PRE: **Subscriptor**.AllInstances()->**exists**(s | **s.usuari = nomS**)

and **Perfil**.AllInstances() -> **exists**(p | **p.nom = nomP** and **p.Subscriptor.usuari = nomS**)

and **Contingut**.AllInstances() -> **exists**(ct | **ct.titol = titolVC**)

and **dataUltimVisionatVC >= Subscriptor**.AllInstances()->**select**(d | **d.nom = nomS).dataAlta**

POST: if not **ContingutVisionat**.AllInstances()**@pre** -> **exists**(cv | **cv.Perfil.nom = nomP**

and **cv.Perfil.Subscriptor.usuari = nomS**

and **cv.Contingut.titol = titolVC**)

then

ContingutVisionat.AllInstances()-> **exists**(cv | **cv.oclIsNew()**

and **cv.dataUltimVisionat = dataUltimVisionatVC**

and **cv.puntuacio = puntuacioVC**

and **cv.comentaris = comentarisVC**

and **cv.Contingut.titol = titolVC**

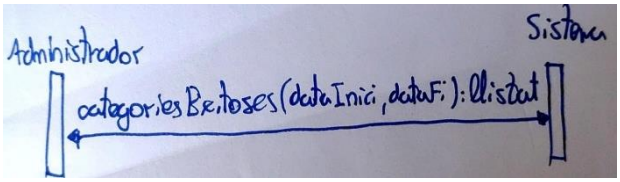
and **cv.Perfil.nom = nomP**

and **cv.Perfil.Subscriptor.usuari =nomS**)

Aquí falta un if ContingutVisionat.AllInstances()@pre -> exists

then lo mismo pero **sin el cv.oclIsNew()**

Categories Exitoses: Quan un administrador vol consultar els continguts més exitosos en un període de temps, indica al sistema les dates d'inici i fi de la consulta. El sistema retorna, per a cada contingut amb més de 5 descàrregues efectuades entre les dues dates (incloses): el títol del contingut, els noms de les seves categories i el preu total acumulat corresponent a totes les seves descàrregues (número de descàrregues * preu). Aquesta funcionalitat només es pot demanar si hi ha més de 3 continguts descarregats entre les dates de la consulta.



CONTEXT: Sistema::**categoriesExitoses(dataInici**:Date**, dataFi**:Date**) :Set(TupleType(titolContigut**:String, **categories**:Set(String), **preuHistoric**:Integer))

PRE: **ContigutDescarregat**.allInstances()->**select**(c| **c.Data.date >= dataInici** AND **c.Data.date <= dataFi) -> size()>=3**

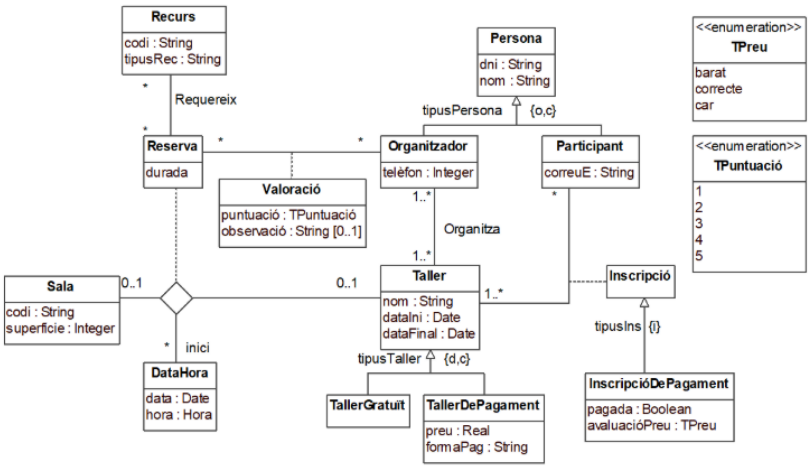
BODY: **l**et llistat:**set(ContigutPremium)** = **ContigutPremium**.allInstances() -> **select**(cd | **cd.titol** -> **select**(c | **c.Data.date >= dataIni**

and **d.Data.date <= dataFi) -> size() >= 5**)

in

result = llistat -> **collect**(cd | **Tuple**{ **titolContingut = cd.titol,** **preuHistoric = cd.preu * cd.ContigutDescarregat ->size(),**

categories = cd.Categoria.nom })

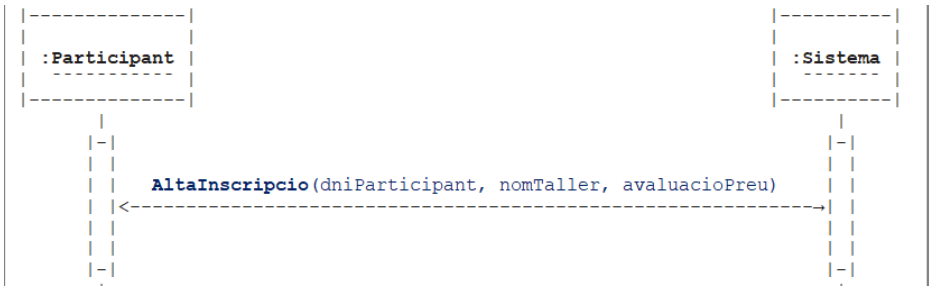


Restriccions Textuals

1. Claus externes: (Taller, nom), (Persona, dni), (Sala, codi), (DataHora, data+hora), (Recurs, codi)
2. La data d'inici d'un taller és anterior o igual a la seva data de finalització
3. Un organitzador d'un taller no pot ser participant d'aquell mateix taller
4. Una inscripció és de pagament si i només si correspon a un taller de pagament
5. Les reserves d'una mateixa sala no es poden solapar temporalment
6. Les reserves d'un mateix taller no es poden solapar temporalment
7. Una reserva ha de tenir una data que estigui entre la data d'inici i de finalització del taller de la reserva
8. Un recurs no pot ser requerit per dues o més reserves que se solapin temporalment
9. Un organitzador no pot valorar una reserva que correspon a un taller que ell no organitza

El sistema a desenvolupar no pot modificar les dades de Persona, Organitzador, Participant, Taller, DataHora, Sala i Valoració, ja que ho fa un altre sistema. El sistema ha de permetre efectuar les següents funcionalitats:

Alta d'Inscripció: Quan un participant vol inscriure's a un taller ell mateix introdueix el seu dni i el nom del taller al que vol inscriure's. Si el taller és de pagament, per defecte es considerarà inscripció no pagada i el participant haurà d'indicar l'avaluació que fa del seu preu. Considereu que una inscripció només es pot fer abans del començament del taller. Feu que la interacció necessària per dur a terme aquesta funcionalitat requereixi només un esdeveniment.



Context: Sistema::**AltaInscripcio** (**dniParticipant**: String, **nomTaller**: String, **avaluacioPreu**: TPreu)

Pre:

Participant.allInstances()->**exists**(p | **p.dni = dniParticipant**)

Taller.allInstances()->**exists**(t | **t.nom = nomTaller**)

Post:

Inscripcio.allInstances->**exists**(i | **i.oclIsNew()**

and **i.persona.dni = dniParticipant**

and **i.taller.nom = nomTaller**

and if **Taller**.allInstances()->**exists**(t |**t.nom = nomTaller** AND **t.oclIsTypeOf(TallerDePagament)**

then

i.oclIsTypeOf(InscripcioDePagament)

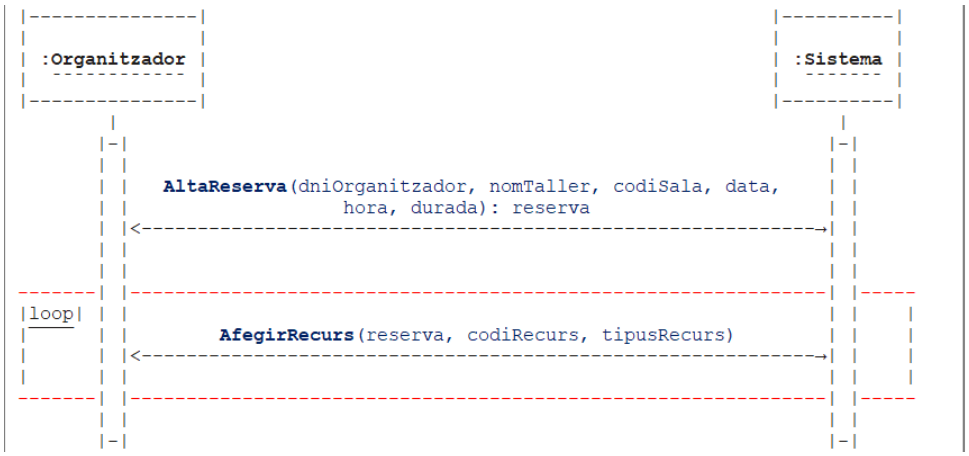
and **i.oclAsType(InscripcioDePagament).pagada = false**

and **i.oclAsType(InscripcioDePagament).avaluacioPreu = avaluacioPreu**

endif

)

Alta de Reserva: Quan un organitzador d'un taller vol fer una reserva de sala, introdueix al sistema les dades necessàries per fer-ho. Per cada recurs que es requereixi a la reserva, haurà d'informar-lo. Si el recurs és nou s'haurà de donar d'alta. Feu que la interacció necessària per dur a terme aquesta funcionalitat requereixi més d'un esdeveniment.



Context: Sistema::**AltaReserva** (**dniOrganitzador**: String, **nomTaller**: String, **codiSala**: String, **data**: Date, **hora**: Hora, **durada**: int): ***Reserva***

Pre:

Persona.allInstances()->**exists**(p | **p.dni = dniOrganitzador**)

Taller.allInstances()->**exists**(t | **t.nom = nomTaller**)

Sala.allInstances()->**exists**(s | **s.codi = codiSala**)

DataHora.allInstances()->**exists**(dh | **dh.data = data** AND **dh.hora = hora**)

Post:

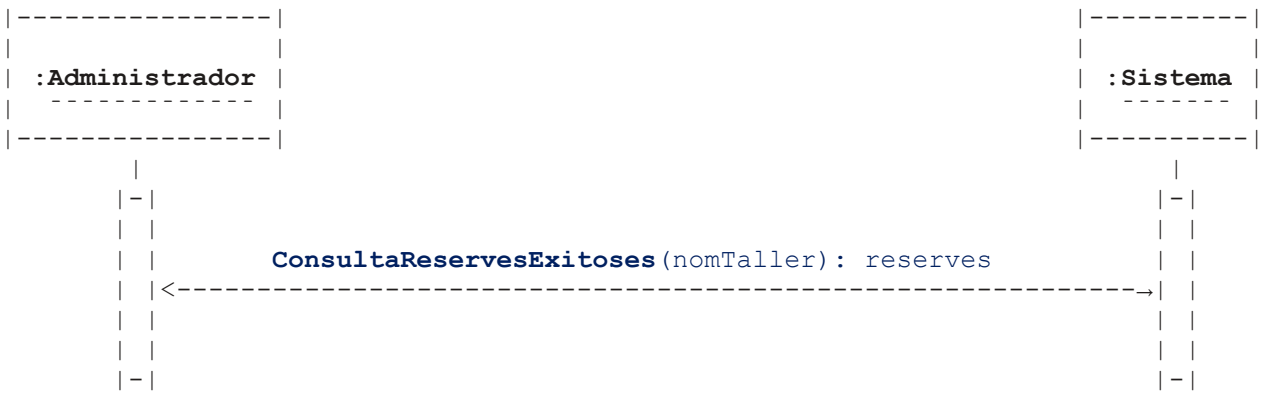
Reserva.allInstances()->**exists**(r | **r.oclIsNew()**
and **r.sala.codi = codiSala**
and **r.dataHora.data = data**
and **r.dataHora.hora = hora**
and **r.taller.nom = nomTaller**
and **r.Organitzador.dni = dniOrganitzador**
and **r.durada = durada**
and **result = r**)

Context: Sistema::**AfegirRecurs** (**reserva**: Reserva, **codiRecurs**: String, **tipusRecurs**: String)

Pre:

Post: if not **Recurs**.allInstances()->**exists**(r | **r.codi = codiRecurs**) then
Recurs.allInstances()->exists(r | **r.oclIsNew()**
AND **r.codi = codiRecurs**
AND **r.tipusRecurs = tipusRecurs**)
endif
AND **Recurs**.allInstances()->**exists**(r | **r.codi = codiRecurs** AND **r.reserva = reserva**)

Consulta Reserves Exitoses: Quan un administrador vol consultar les reserves més exitoses d'un taller de pagament indica al sistema el nom del taller. El sistema retorna, per a cada reserva amb més de 5 valoracions amb puntuació de 5 d'aquest taller: el codi de la sala reservada, la data i hora de la reserva i el conjunt de correus de participants inscrits. Aquesta funcionalitat solament pot demanar-se si hi ha com a mínim 3 inscripcions al taller pagades i amb avaluació de barat.



Context: Sistema:**ConsultaReservesExitoses** (**nomTaller**: String): Set(TupleType (**codiSala**: String, **data**: String, **hora**: String, **correusParticipants**: Set(String)))

Pre:

```
InscripcioDePagament.allInstances()->select(i | i.taller.nom = nomTaller
                                            and i.taller.oclIsTypeOf(TallerDePagament)
                                            and i.pagada
                                            and i.avaluacioPreu = TPreu::barat )->size() >= 3
```

Body:

```
let reserves: Set(Reserva) = Reserva.allInstances()->select(r | r.taller.nom = nomTaller
                                                        and r.valoracio->select(v | v.puntuacio = TPuntuacio::5
                                                                                and v.reserva = r )->size() > 5)
in
 reserves->collect(r |Tuple { codiSala = r.ala.codi,
                              data = r.dataHora.data,
                              hora = r.dataHora.hora,
                             correusParticipants = r.taller.participants.correuE } )
```