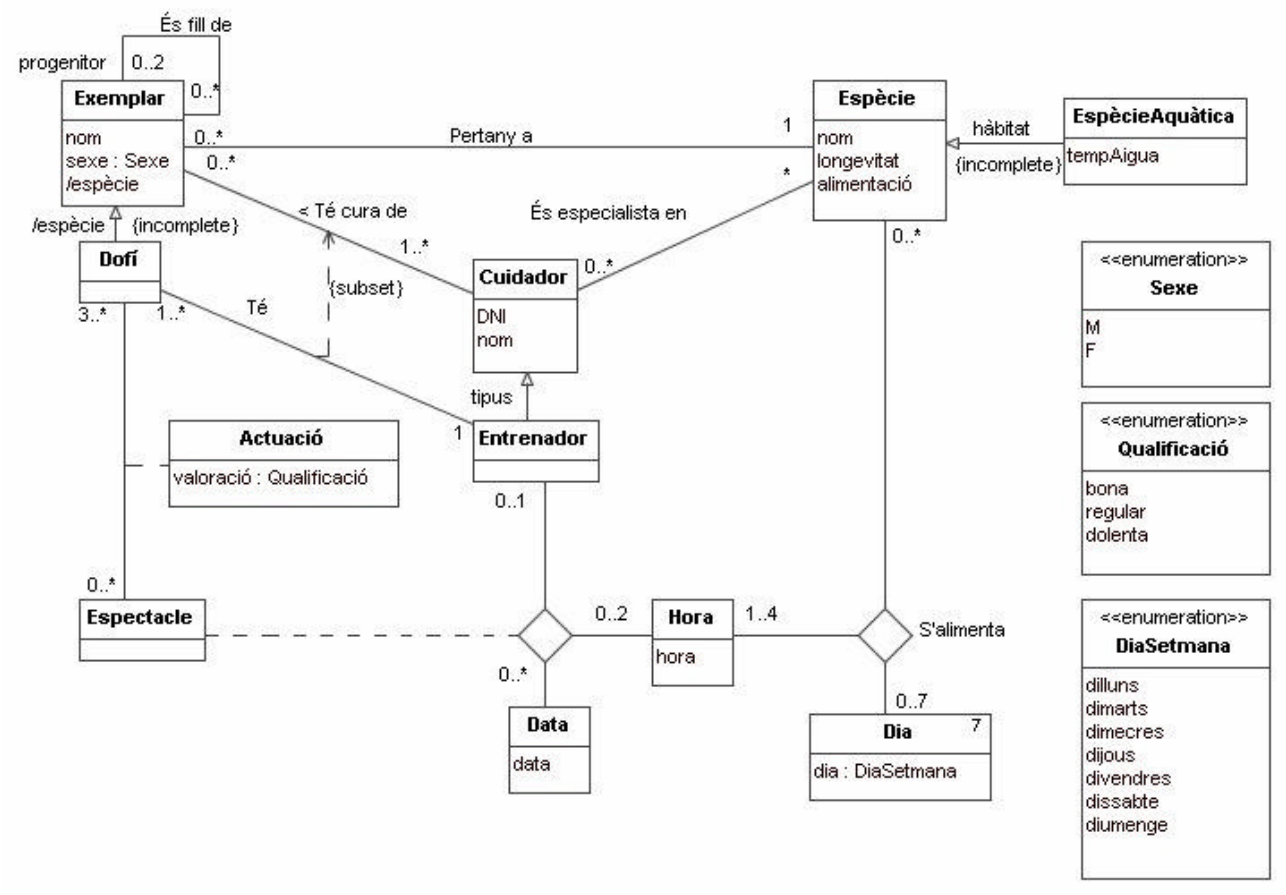
1er Problema – Model Conceptual de les Dades (3,5 punts)



Restriccions textuals:

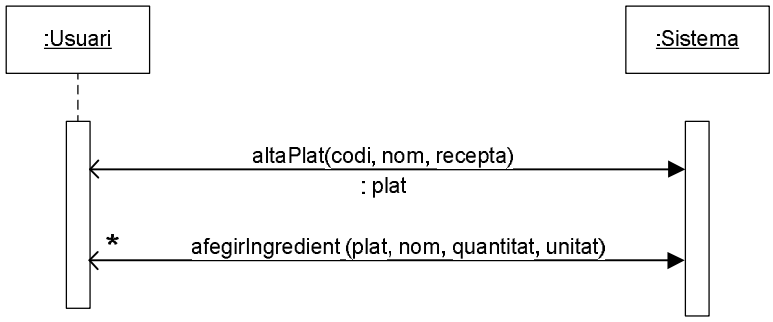
1- Claus externes: (Espècie: nom), (Cuidador: DNI), (Data: data), (Hora:hora), (Dia: dia).
2- Una espècie no pot tenir dos exemplars amb el mateix nom.
3- L'associació 'Es fill de' no té cicles.
4- Si un exemplar té dos progenitors, aquests són de sexe diferent.
5- Els progenitors d'un exemplar són de la mateixa espècie que l'exemplar fill.
6- Un cuidador no pot tenir cura d'exemplars d'una espècie en la que no és especialista.

Regla de derivació:

1- L'espècie d'un exemplar és el nom de l'espècie a la que pertany l'exemplar.

2on Problema – Model del Comportament (3,5 punts)

Cas d'us AltaPlatIndividual



Operació: `altaPlat(codi:String, nom:String, recepta:String): PlatIndividual`

Semàntica: Enregistra un nou plat individual amb les dades indicades

Precondició: -

Postcondició: `p.oclIsNew() and p.oclIsTypeOf(PlatIndividual) and p.codi=codi and p.nom=nom and p.recepta=recepta`

Sortida: `result = p`

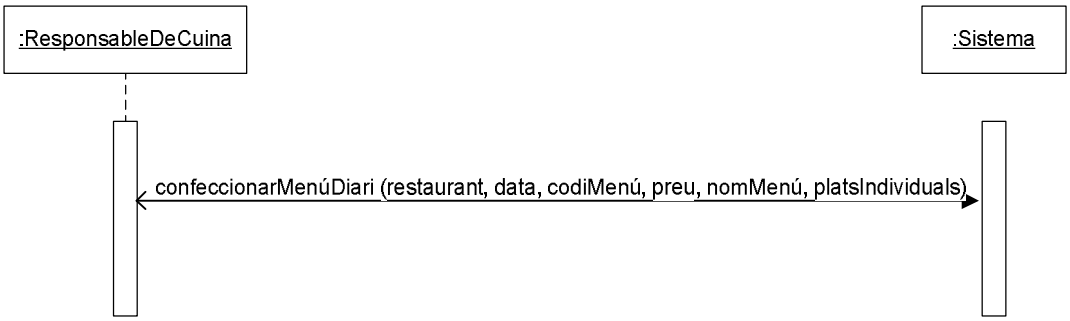
Operació: `afegirIngredient(plat:PlatIndividual, nom:String, quantitat:Decimal, unitat:String): PlatIndividual`

Semàntica: Afegeix un ingredient al plat, la quantitat i unitat de mesura del mateix dins del plat i el conjunt dels possibles ingredients substituts que pot tenir

Precondició:
- `Ingredient.allInstances()->exists(i|i.nom=nom)`
- `not plat.ingredient->exists(i|i.nom=nom)`

Postcondició: `m.oclIsNew() and m.oclIsTypeOf(Mesura) and m.quantitat=quantitat and m.unitatDeMesura=unitat and m.platIndividual=plat and m.ingredient.nom=nom`

Cas d'us ConfeccióMenúDiari



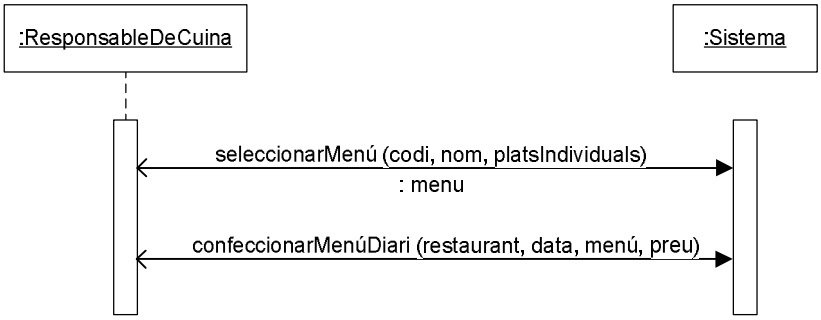
Operació: `confeccionarMenú(restaurant:String, data:Date, codiMenú:String, preu:Decimal, nomMenú:String, platsIndividuals:Set(String))`

Semàntica: Crea un menú diari ofert a partir dels paràmetres que se li passen, comprovant prèviament que existeix un menú i si no és el cas, es dona d'alta.

Precondició:
```
- PlatIndividual.allInstances().codi->
includesAll(platsIndividuals)
- Restaurant.allInstances()->exists(r|r.nom=restaurant)
- not MenúDiariOfert.allInstances()->exists(m|m.menú = menú
and m.restaurant.nom = nom and m.data.data=data)
```

Postcondició:
```
if not Menú.allInstances()-> exists (m|m.codi=codiMenú) then
me.oclIsNew() and me.oclIsTypeOf(Menú) and me.codi=codiMenú
and me.nom=nom and me.platIndividual.codi->
includesAll(platsIndividuals) endif and
m.oclIsNew() and m.oclIsTypeOf(MenúDiariOfert) and
m.restaurant.nom=restaurant and m.data.data=data and
m.menú.codi=codiMenú and m.preu=preu
```

ALTERNATIVA: amb dos operacions



Operació: `seleccionarMenú(codi:String, nom:String, platsIndividuals:Set(String)): menú`

Semàntica: Comprova que existeixi un menú i si no és el cas, el dona d'alta. Els paràmetres nom i platsIndividuals sólament han d'estar informats si no existeix el menú.

Precondició:
```
PlatIndividual.allInstances().codi->
includesAll(platsIndividuals)
```

Postcondició:
```
if not Menú.allInstances()-> exists (m|m.codi=codi) then
me.oclIsNew() and me.oclIsTypeOf(Menú) and me.codi=codi and
me.nom=nom and me.platIndividual.codi->
includesAll(platsIndividuals) endif
```

Sortida: `result = Menú.allInstances()->select(m|m.codi=codi)`

Operació: `confeccionarMenúDiari(restaurant:String, data:Date, menú:Menú, preu:Decimal)`

Semàntica: Es crea un menú diari ofert a partir dels paràmetres que se li passen

Precondició: `- Restaurant.allInstances()->exists(r|r.nom=restaurant)`

```
- not MenúDiariOfert.allInstances()->exists(m|m.menú = menú
and m.restaurant.nom = nom and m.data.data=data)
```

Postcondició:
```
m.oclIsNew() and m.oclIsTypeOf(MenúDiariOfert) and
m.restaurant.nom=restaurant and and m.data.data=data and
m.menú=menú and m.preu=preu
```

Cas d'us ConsultaPlatsOferts

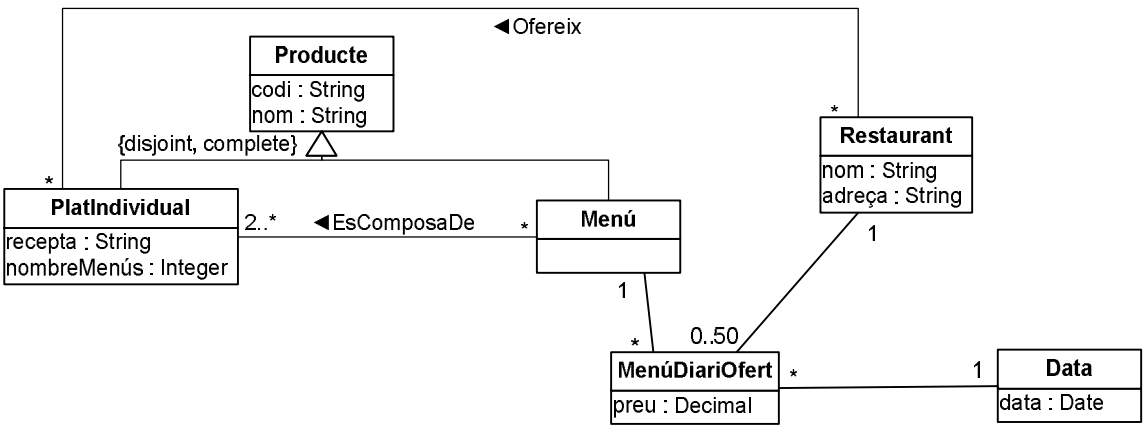
Operació:
```
consultaPlats(dataInici:Date, dataFi:Date, restaurant:String)
: Set(TupleType(codiPlat:String, nomPlat:String,
ingredients:Set (TupleType(nom:String, quantitat:Decimal,
unitat:String))))
```

Semàntica: Retorna un llistat de tots els plats que s'han ofert en un restaurant en un periode i per cadascun d'ells, els ingredients i la quantitat i mesura que se n'usa.

Precondició: (opcionals les dos)
```
- dataFi=dataInici
- Restaurant.allInstances()->exists(r| r.nom=restaurant)
```

Sortida:
```
let platsOferts: Set(PlatIndividual) =
MenúDiariOfert.allInstances()->select (m|
m.data.data>=dataInici and m.data.data<=dataFi and
m.restaurant.nom = restaurant).menú.platIndividual->asSet()
in
result = platsOferts->collect(p| Tuple{codiPlat=p.codi,
nomPlat=p.nom, ingredients = p.mesura-> collect(me |
Tuple{nom=me.ingredient.nom, quantitat=me.quantitat,
unitat=me.unitatDeMesura})})
```

NORMALITZACIÓ



RI Textuals afegides:
- No hi pot haver dos menúDiariOfert amb els mateixos Restaurant, Data i Menú.
- Un restaurant pot oferir com a màxim 5 menús en una dia.
- S'elimina la segona restricció ja que ara és gràfica.

Contracte confeccionarMenúDiari:

afegir a la precondició:
- Cap dels MenúDiariOfert té el mateix nom de Restaurant, Data i codi de Menú associat (control de la primera restricció afegida).
- El restaurant no pot tenir en la mateixa data ja 5 menús diferents associats (control de la segona restricció afegida)
- El restaurant no pot tenir en total ja 50 diferents menúDiariOfert (control de la cardinalitat gràfica 0..50)
- Si no existeix el menú, el paràmetre platsIndividuals conté com a mínim 2 codis (control de la cardinalitat 2..*)

afegir a la postcondició:
- Si s'ha creat un nou menú, per cada platIndividual inclòs en el menú s'afegeix l'associació restaurant ofereix platIndividual (si no existia prèviament)
- Si s'ha creat un nou menú, a cada platIndividual inclòs en el menú s'incrementa en 1 el nombreMenús.

La resta queda igual.