PRÀCTICA ARQUITECTURA DEL SOFTWARE

LLIURAMENT 1

Robert Almar Graupera

Daniel García Romero

Alvaro Martínez Arroyo

Alex Vilarrubla Martín

1.

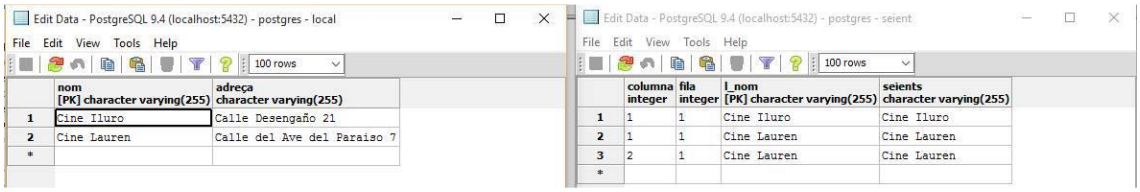
Joc de proves

```
package as;

import java.util.ArrayList;
import java.util.Iterator;
import javax.imageio.spi.ServiceRegistry;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.boot.registry.StandardServiceRegistry;
import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
import org.hibernate.cfg.Configuration;
import org.hibernate.tool.hbm2ddl.SchemaExport;

/**
 *
 * @author rober_000
 */
public class AS {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        Configuration config = new Configuration();
        config.addAnnotatedClass(Seient.class);
        config.addAnnotatedClass(Local.class);
        config.configure("hibernate.cfg.xml");

        new SchemaExport(config).create(true, true);


        StandardServiceRegistry serviceRegistry = new StandardServiceRegistryBuilder().applySettings(config.getProperties()).build();

        SessionFactory factory = config.buildSessionFactory(serviceRegistry);
        Session session = factory.openSession();
        session.beginTransaction();

        ArrayList<Seient> s = new ArrayList<>();
        ArrayList<Seient> s2 = new ArrayList<>();

        Local l = new Local();
        l.setAdreca("Calle Desengaño 21");
        l.setNom("Cine Iluro");
        l.setSeients(s);

        Local l2 = new Local();
        l2.setAdreca("Calle del Ave del Paraiso 7");
        l2.setNom("Cine Lauren");
        l2.setSeients(s2);

        CompoundKeySeient CKS = new CompoundKeySeient();
        CKS.setColumna(1);
        CKS.setFila(1);
        CKS.setLocal(l);

        CompoundKeySeient CKS2 = new CompoundKeySeient();
        CKS2.setColumna(1);
        CKS2.setFila(1);
        CKS2.setLocal(l2);

        CompoundKeySeient CKS3 = new CompoundKeySeient();
        CKS3.setColumna(2);
        CKS3.setFila(1);
        CKS3.setLocal(l2);

        Seient seient = new Seient();
        seient.setCompoundKeySeient(CKS);
        l.setSeient(seient);

        Seient seient2 = new Seient();
        seient2.setCompoundKeySeient(CKS2);
        l2.setSeient(seient2);

        Seient seient3 = new Seient();
        seient3.setCompoundKeySeient(CKS3);
        l2.setSeient(seient3);

        session.save(l);
        session.save(l2);
        session.save(seient);
        session.save(seient2);
        session.save(seient3);

        session.getTransaction().commit();

        System.out.println("GET ADREÇA DEL LOCAL L2: " + l2.getAdreca());
        System.out.println("GET NOM DEL LOCAL L2: " + l2.getNom());
        System.out.println("GET SEIENTS DEL LOCAL L2:");
        Iterator<Seient> it = l2.getTotsSeients().iterator();
        int i = 1;
        while(it.hasNext()){
            System.out.println("Seient " + i);
            Seient s1 = it.next();
            System.out.println("LOCAL DEL SEIENT: " + s1.getCompoundKeySeient().getLocal().getNom() + " FILA: " + s1.getCompoundKeySeient().getFila() + " COLUMNA: " + s1.getCompoundKeySeient().getColumna());
            ++i;
        }

        }
}
```

Sortida consola

GET ADREÇA DEL LOCAL L2: Calle del Ave del Paraiso 7
GET NOM DEL LOCAL L2: Cine Lauren
GET SEIENTS DEL LOCAL L2:
Seient 1
LOCAL DEL SEIENT: Cine Lauren FILA: 1 COLUMNA: 1
Seient 2
LOCAL DEL SEIENT: Cine Lauren FILA: 1 COLUMNA: 2

Sortida Postgres



Local.java

```java
package as;

import java.util.ArrayList;
import java.util.Collection;
import java.util.HashSet;
import java.util.Set;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.Table;
import javax.persistence.OneToMany;
/**
 *
 * @author rober_000
 */
@Entity
@Table(name = "LOCAL")
public class Local {
    @Id
    private String nom;
    @Column(name = "Adreça")
    private String adreca;
    @OneToMany
    @JoinColumn(name = "Seients")
    private Collection<Seient> seients;

    public Local(){}
    public Local(String name){
        nom = name;
    }


    public String getNom(){
        return nom;
    }

    public String getAdreca(){
        return adreca;
    }

    public Collection<Seient> getTotsSeients(){
        return seients;
    }

    public void setNom(String name){
        nom = name;
    }

    public void setAdreca(String address){
        adreca = address;
    }

    public void setSeients(Collection<Seient> s){
        seients = s;
    }

    public void setSeient(Seient s){
        seients.add(s);
    }
}
```

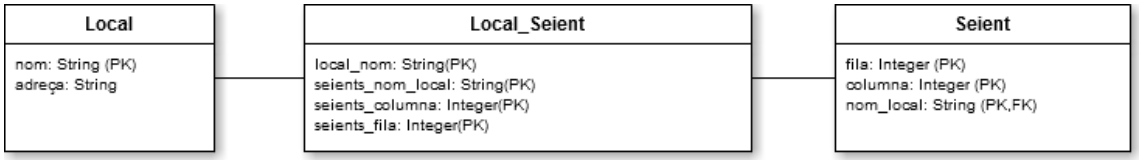
Seient.java

```java
package as;

/**
 *
 * @author rober_000
 */
import java.util.ArrayList;
import javax.persistence.Entity;
import javax.persistence.ForeignKey;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name = "SEIENT")
public class Seient {
    @Id
    private CompoundKeySeient CKS;

    public Seient(){

    }
    public CompoundKeySeient getCompoundKeySeient(){
        return CKS;
    }

    public void setCompoundKeySeient(CompoundKeySeient compoundKeySeient){
        CKS = compoundKeySeient;
    }

}
```

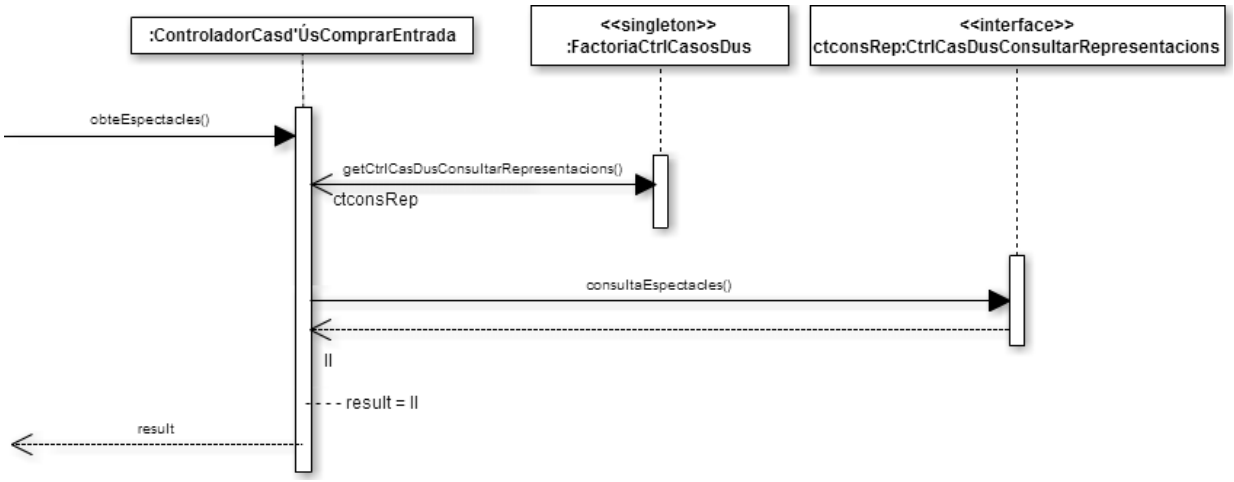
CompoundKeySeient.java

```java
package as;

import java.io.Serializable;
import javax.persistence.Column;
import javax.persistence.Embeddable;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;

/**
 *
 * @author rober_000
 */
@Embeddable
public class CompoundKeySeient implements Serializable{
    @Column(name = "fila")
    private Integer fila;
    @Column(name = "columna")
    private Integer columna;
    @ManyToOne
    //@Column(name = "Nom Local")
    private Local L;

    public CompoundKeySeient(){}

    public CompoundKeySeient(Local loc, Integer row, Integer column){
        L = loc;
        fila = row;
        columna = column;
    }

    public Integer getFila(){
        return fila;
    }

    public Integer getColumna(){
        return columna;
    }

    public Local getLocal(){
        return L;
    }

    public void setFila(Integer row){
        fila = row;
    }

    public void setColumna(Integer column){
        columna = column;
    }

    public void setLocal(Local local){
        L = local;
    }
}
```

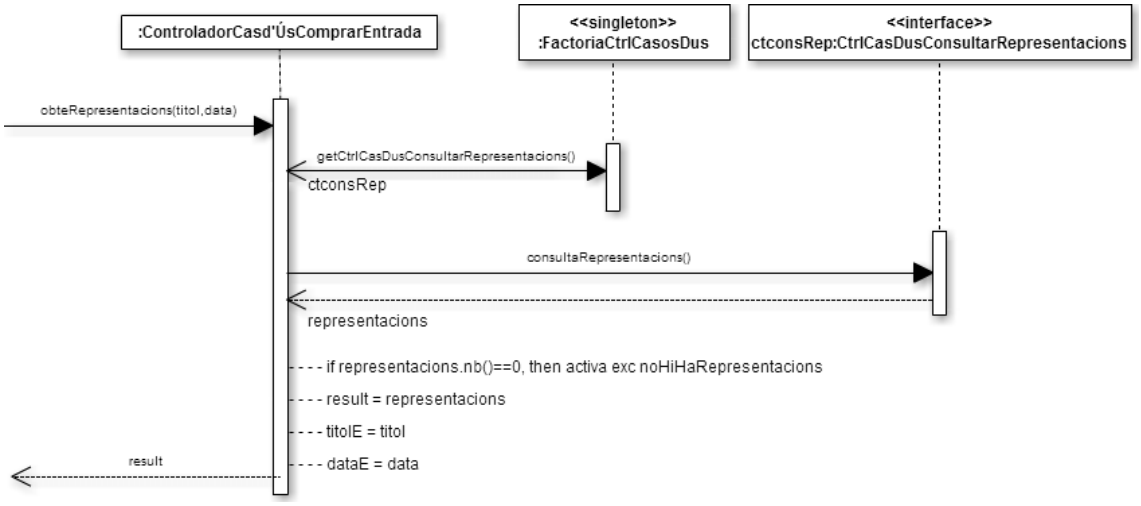
Esquema de la base de dades

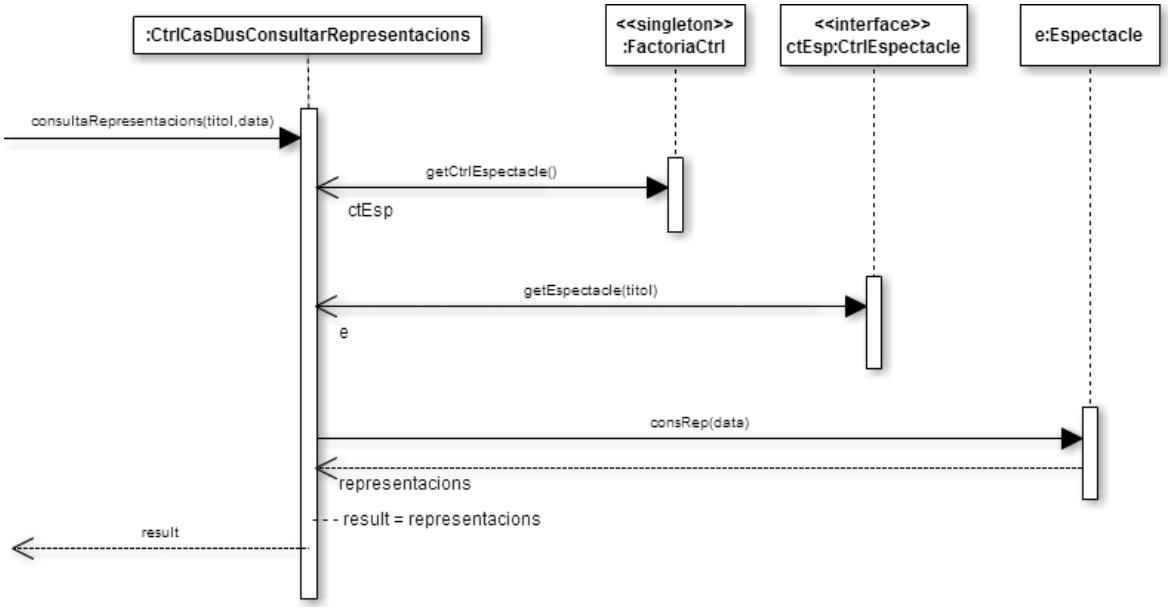


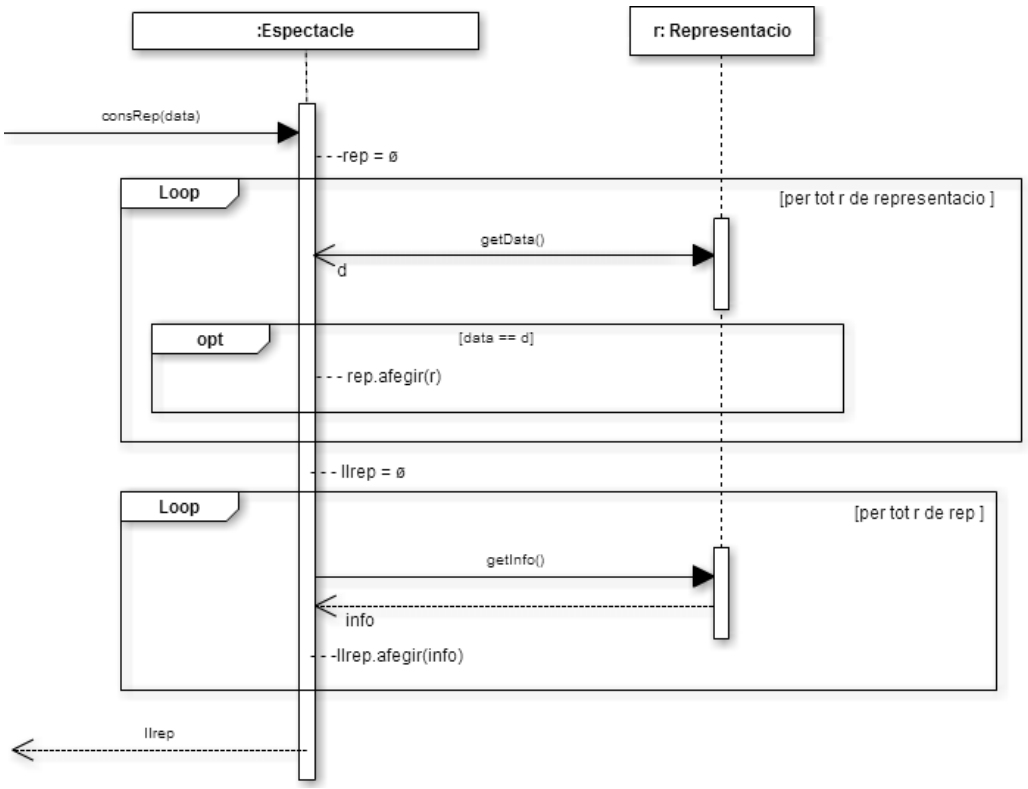
2.



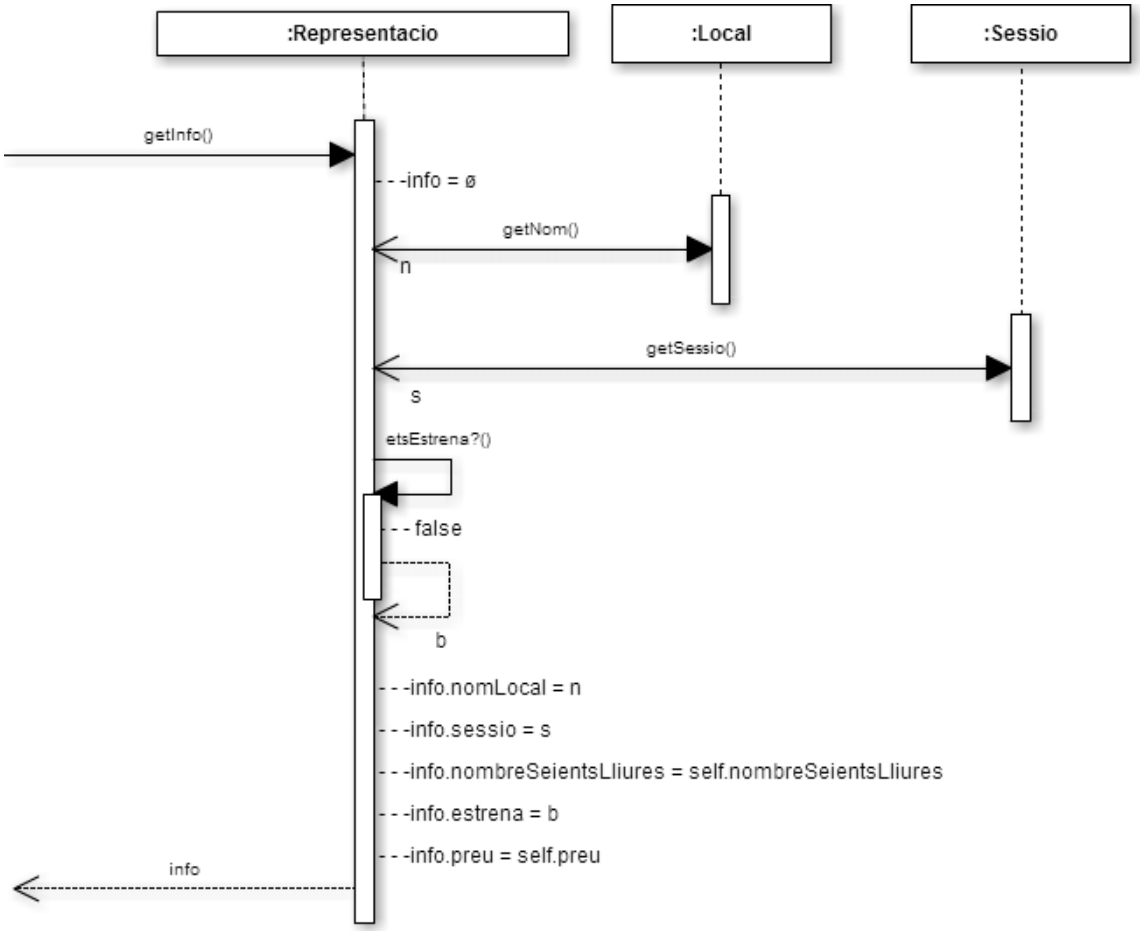
Error1!!! no se deben regresar objetos, sino lista de títulos. Así que hace falta recorrer el conjunto de objetos Espectacle e invocar getTitol.

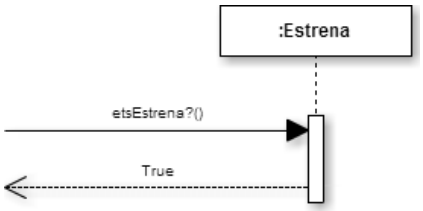


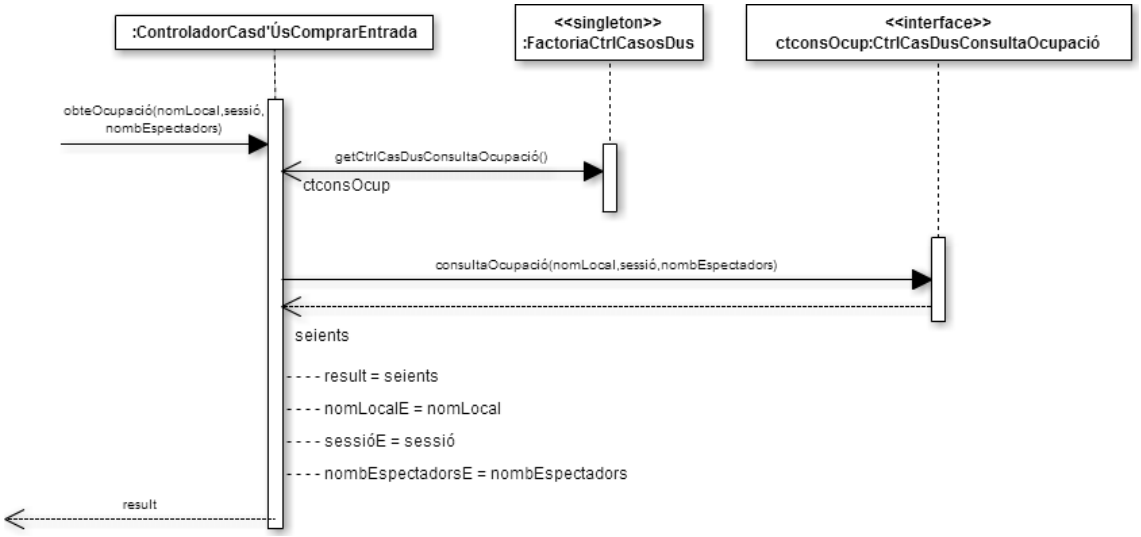


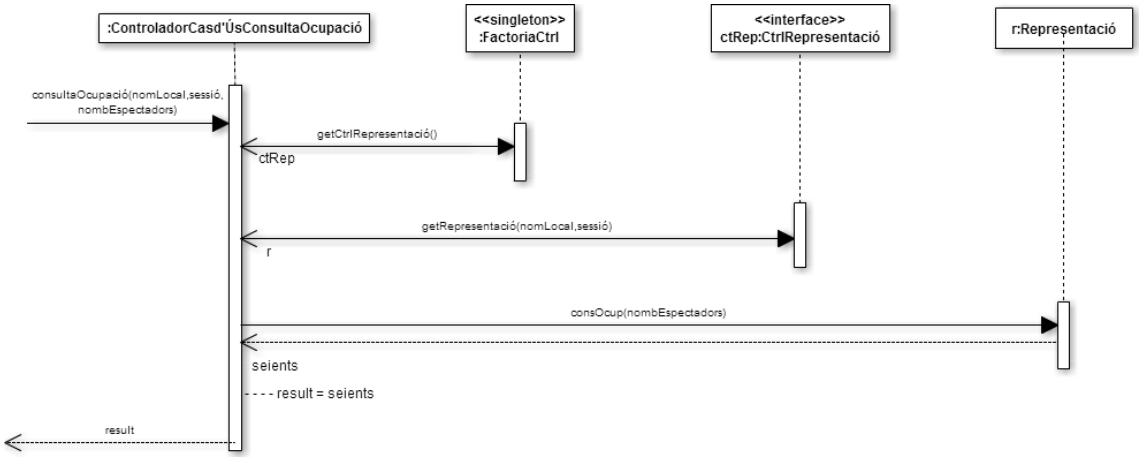


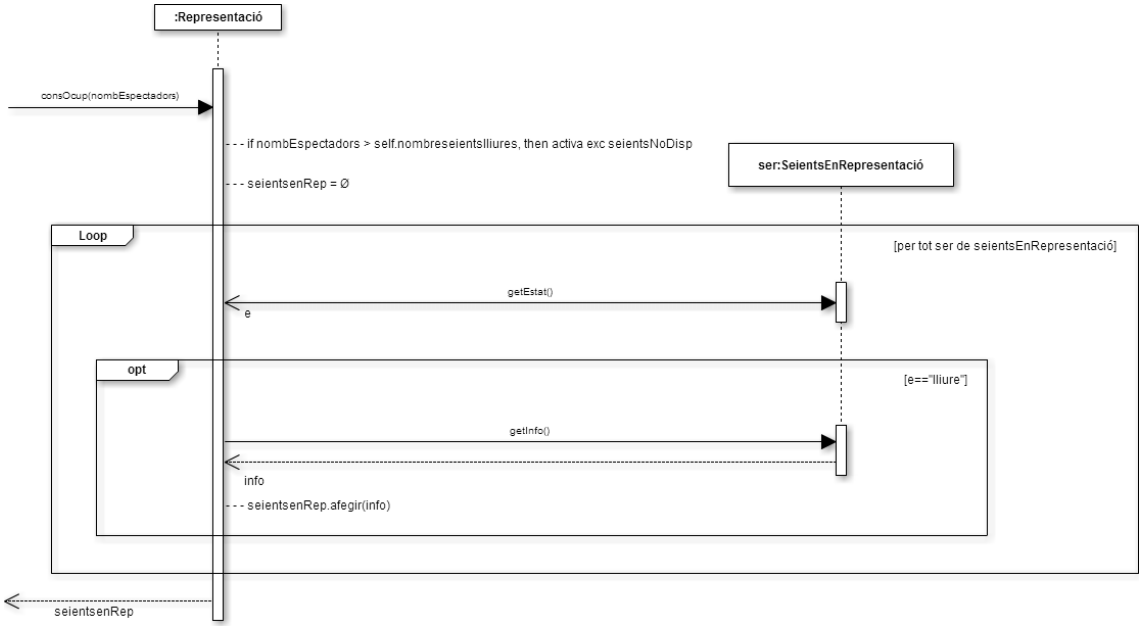
Porqué dos Loops diferentes? Es ineficiente, podría usarse el mismo

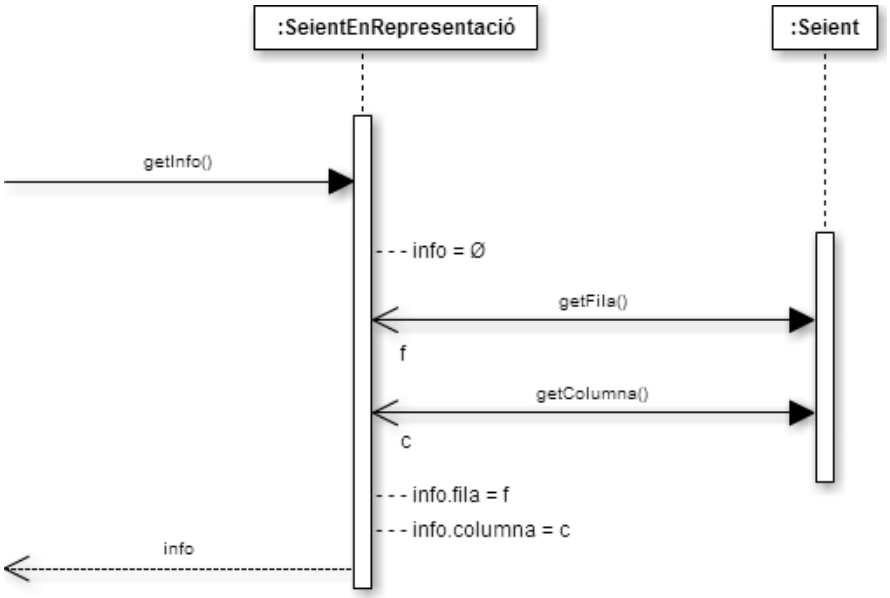

:Representacio
:Local
:Sessio
getInfo()
- -info = ø
getNom()
n
getSessio()
s
etsEstrena?()
- - false
b
- - -info.nomLocal = n
- - -info.sessio = s
- - -info.nombreSeientsLliures = self.nombreSeientsLliures
- - -info.estrena = b
- - -info.preu = self.preu
info

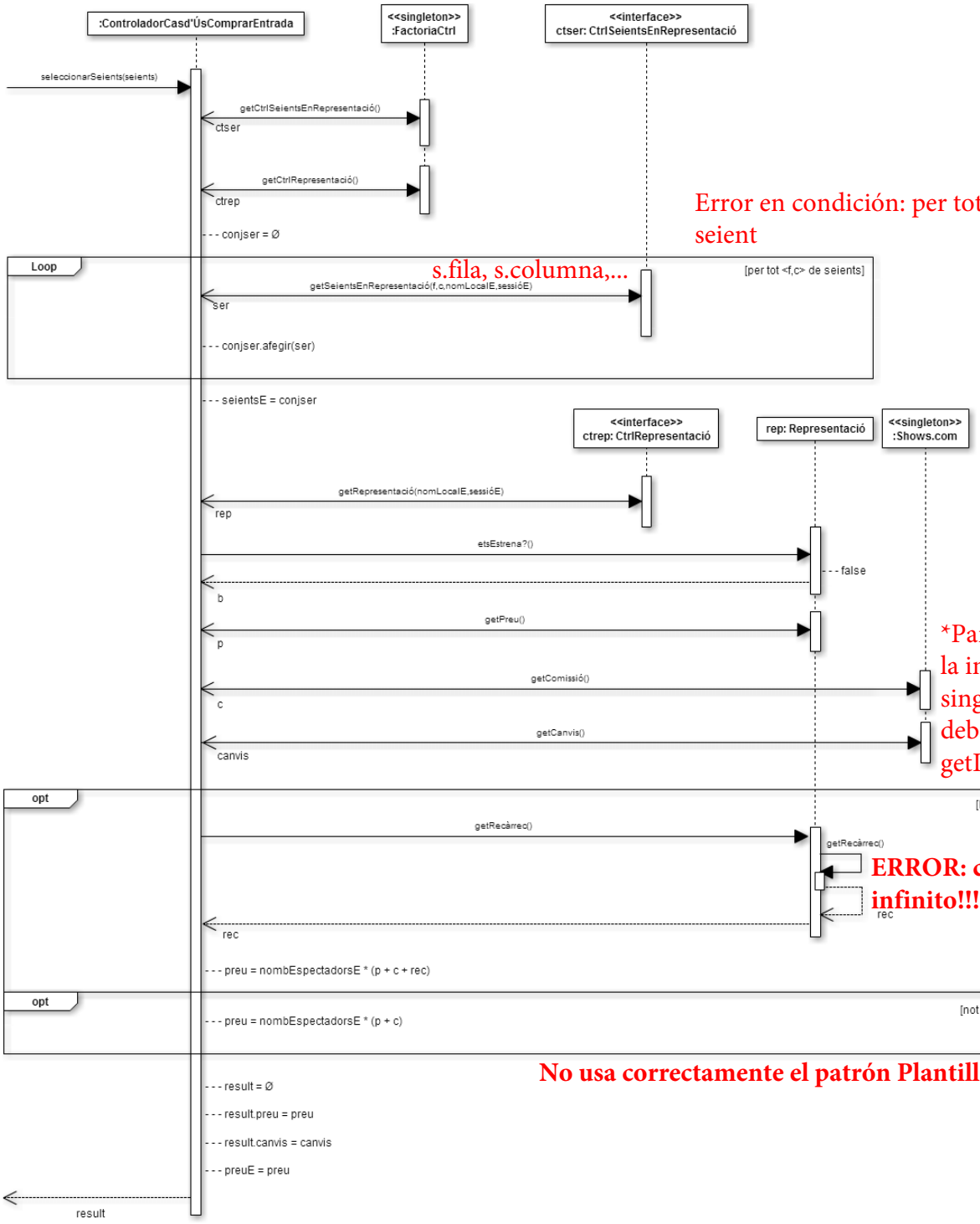

:Estrena
etsEstrena?()
True


:ControladorCasd'ÚsComprarEntrada
<<singleton>>
:FactoriaCtrlCasosDus
<<interface>>
ctconsOcup:CtrlCasDusConsultaOcupació
obteOcupació(nomLocal,sessió,
nombEspectadors)
getCtrlCasDusConsultaOcupació()
ctconsOcup
consultaOcupació(nomLocal,sessió,nombEspectadors)
seients
- - - - result = seients
- - - - nomLocalE = nomLocal
- - - - sessióE = sessió
- - - - nombEspectadorsE = nombEspectadors
result


:ControladorCasd'ÚsConsultaOcupació
<<singleton>>
:FactoriaCtrl
<<interface>>
ctRep:CtrlRepresentació
r:Representació
consultaOcupació(nomLocal,sessió, nombEspectadors)
getCtrlRepresentació()
ctRep
getRepresentació(nomLocal,sessió)
r
consOcup(nombEspectadors)
seients
result = seients
result


:Representació
consOcup(nombEspectadors)
if nombEspectadors > self.nombreseientsIliures, then activa exc seientsNoDisp
seientsenRep = Ø
ser:SeientsEnRepresentació
Loop
[per tot ser de seientsEnRepresentació]
getEstat()
e
opt
[e=="lliure"]
getInfo()
info
seientsenRep.afegir(info)
seientsenRep


:SeientEnRepresentació
:Seient
getInfo()
info = Ø
getFila()
f
getColumna()
c
info.fila = f
info.columna = c
info



Error en condición: per tot s de seient

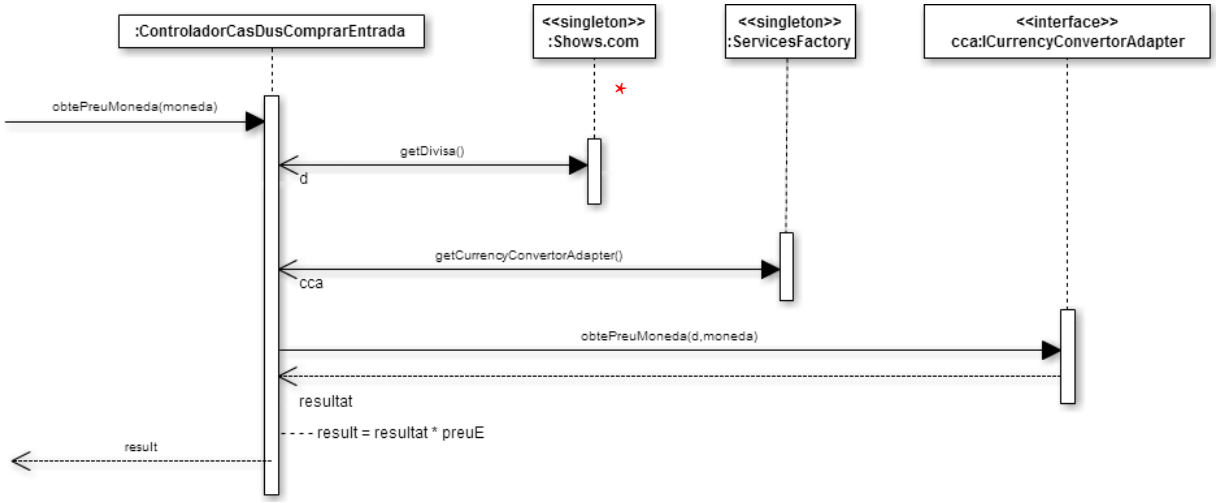
s.fila, s.columna,...

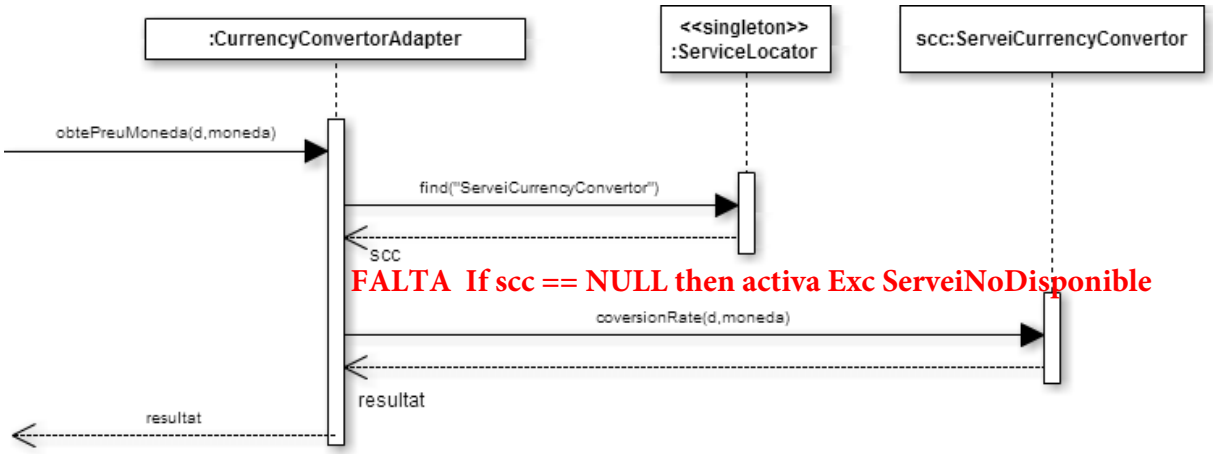
*Para acceder a la instancia singleton se debe hacer un getInstance()

ERROR: ciclo infinito!!!!

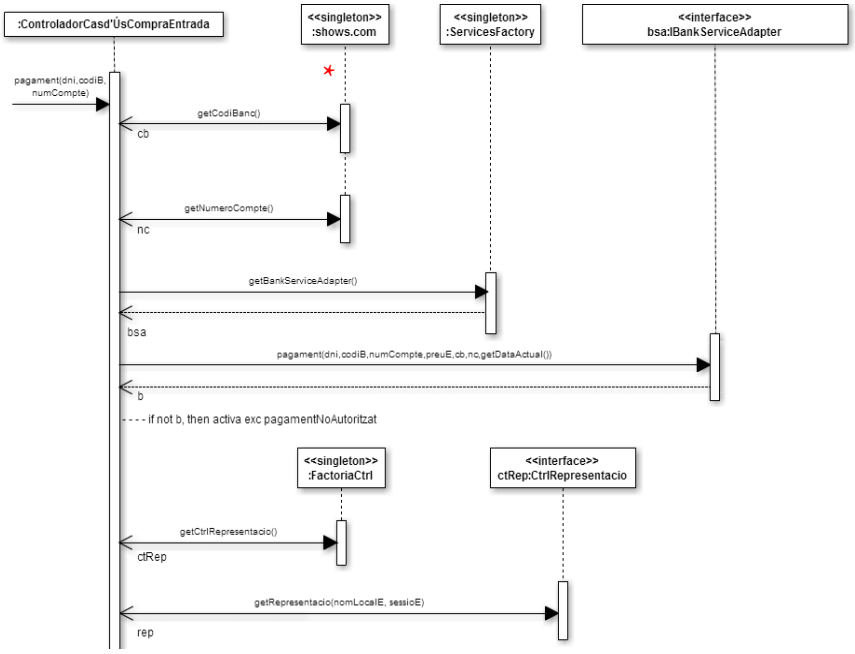
No usa correctamente el patrón Plantilla!

Nombre confuso:mejor AdaptersFactory

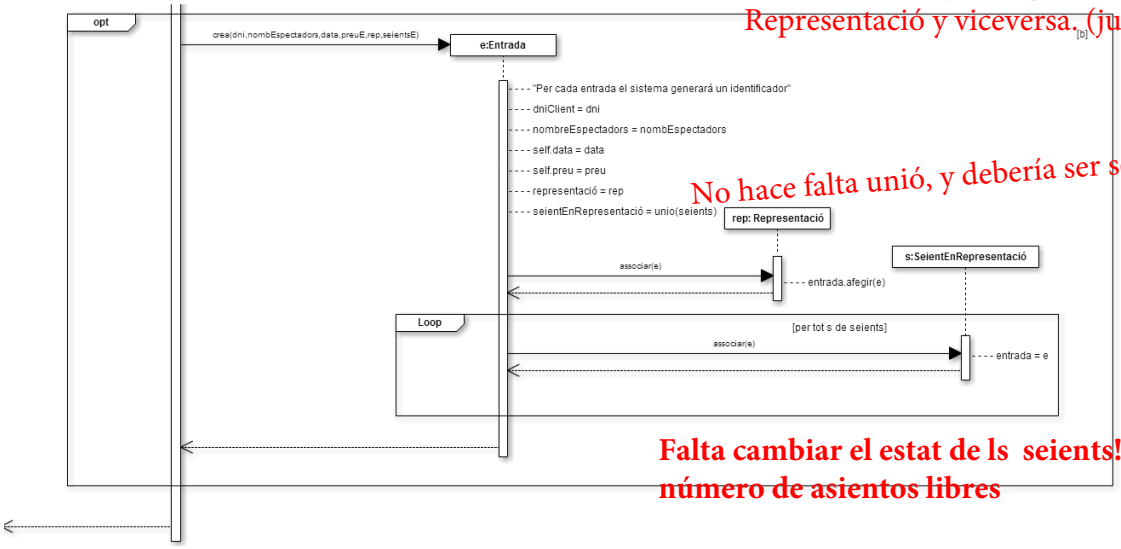




FALTA If scc == NULL then activa Exc ServeiNoDisponible

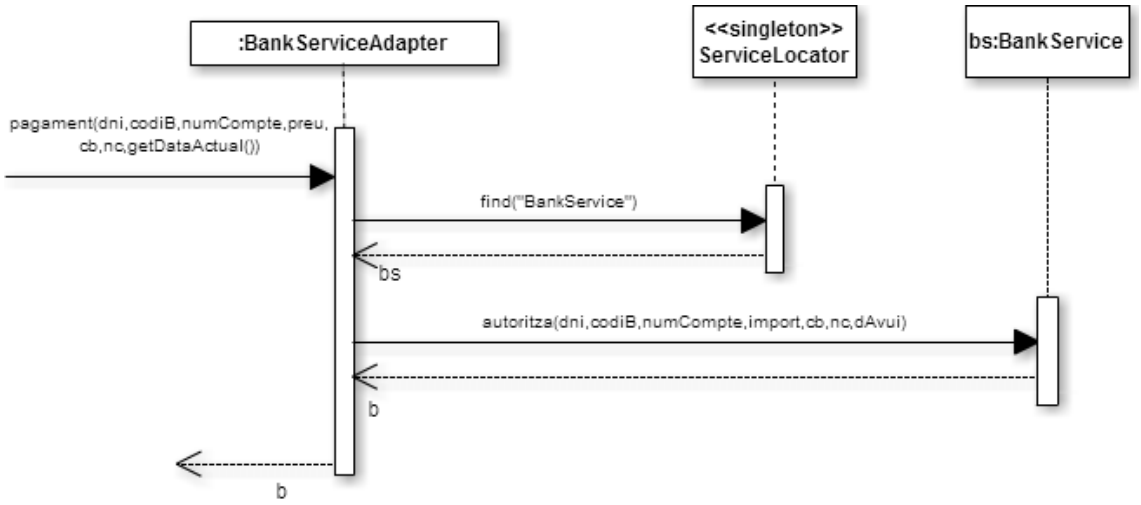


-Asumís que hay navegabilidad de Entrada a Representació y viceversa. (justificar)

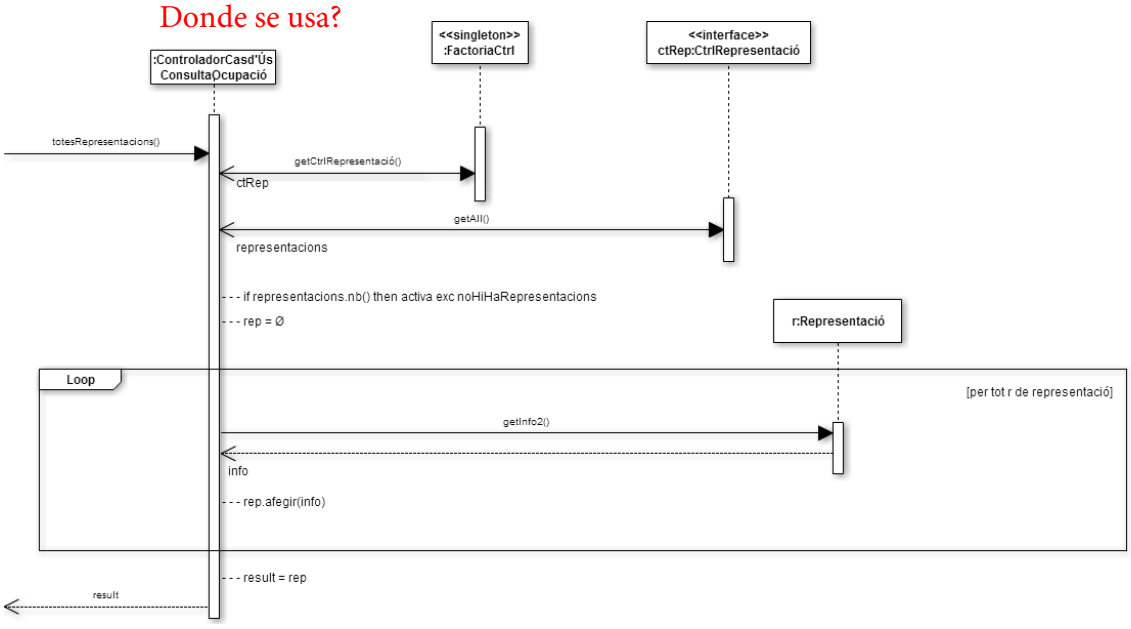


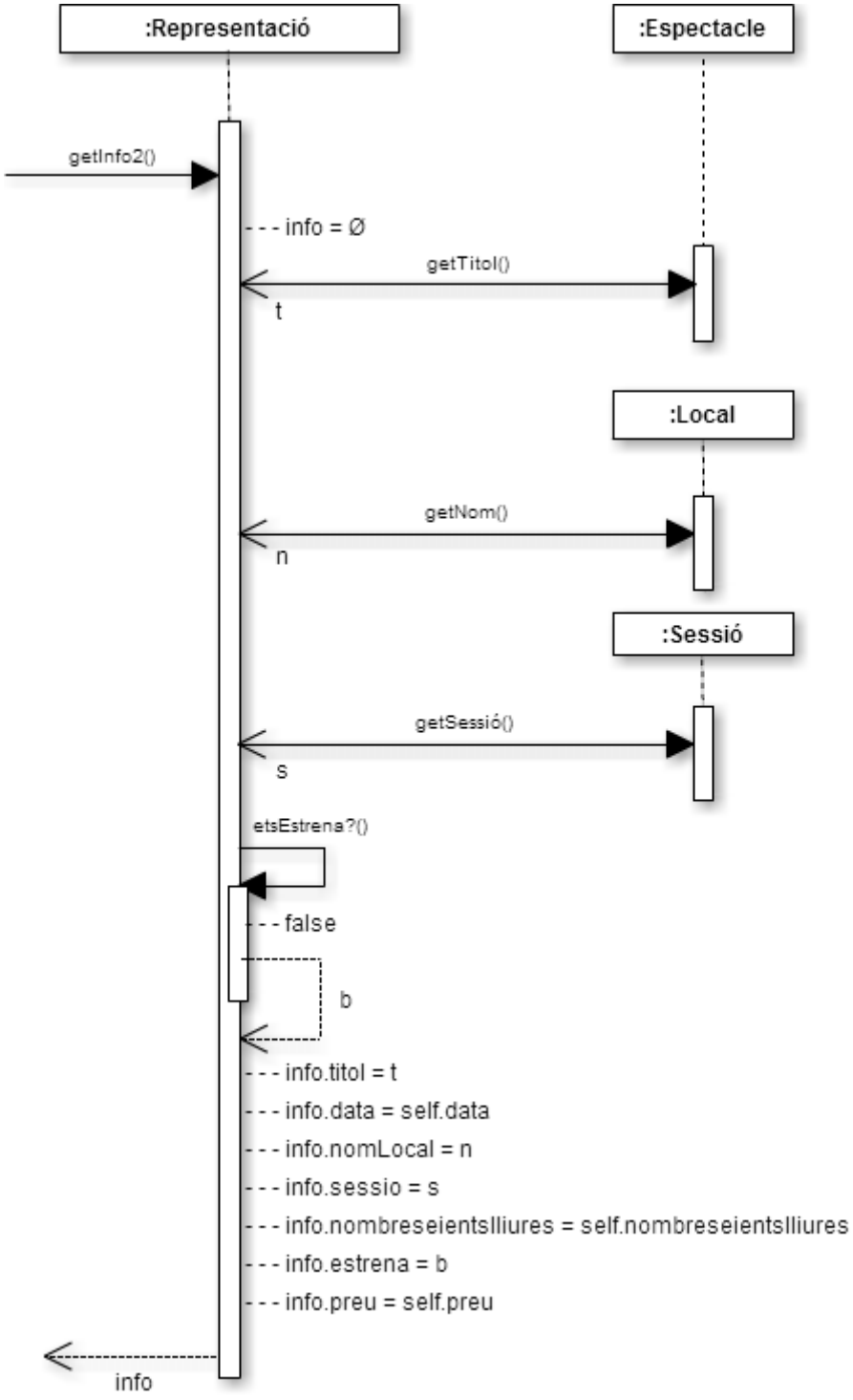
No hace falta unió, y debería ser seientE

Falta cambiar el estat de ls seients! y actualizar número de asientos libres

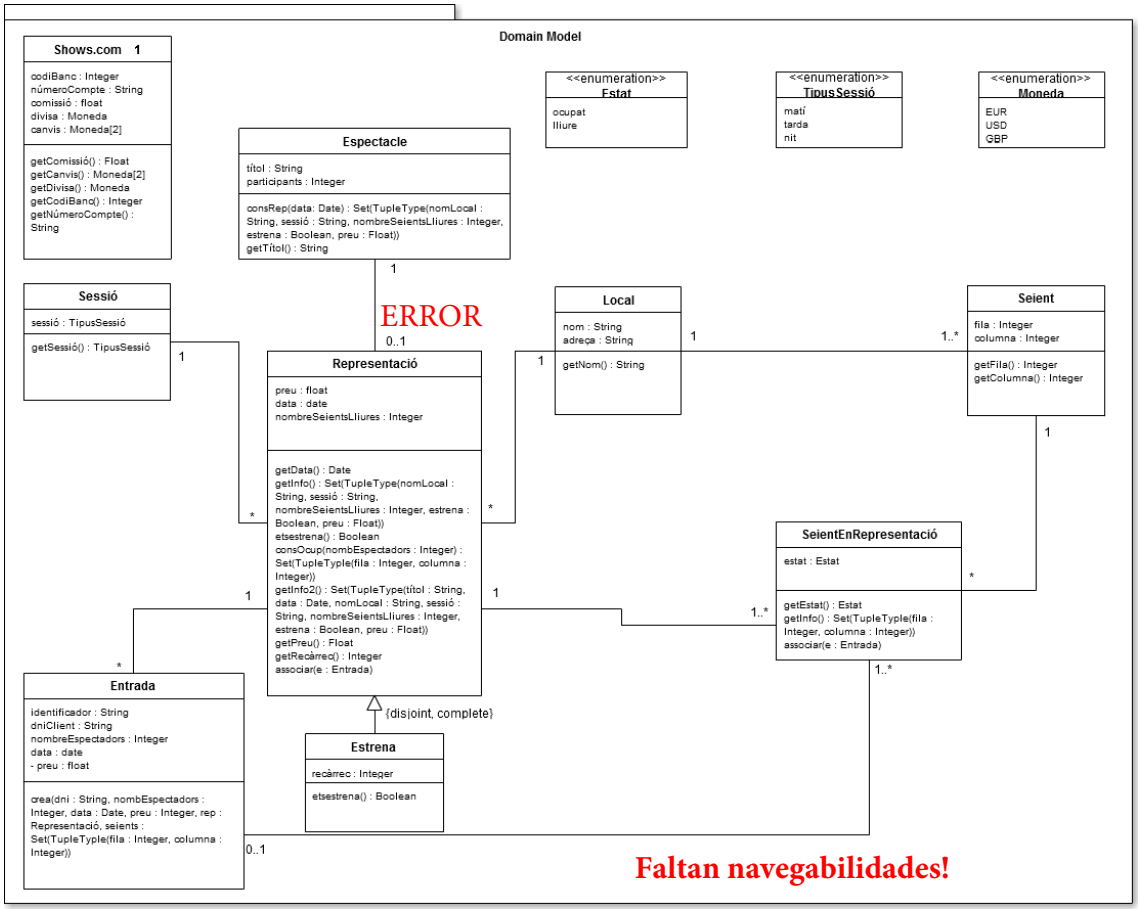


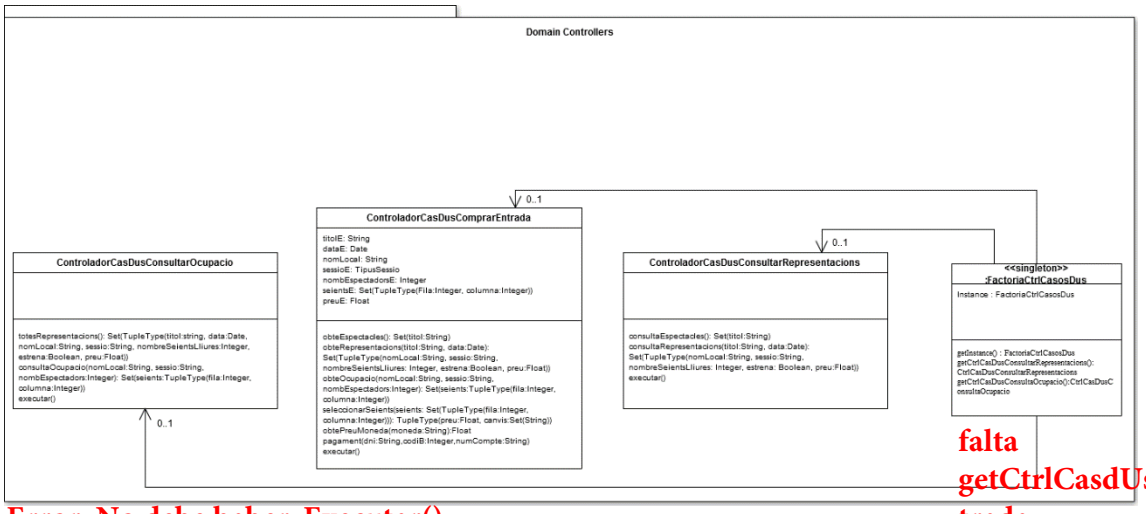
Donde se usa?

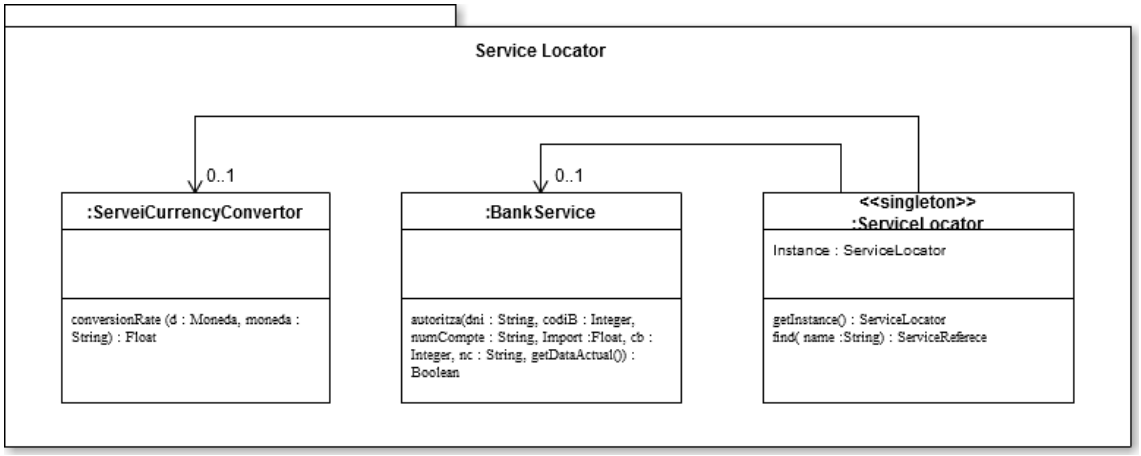


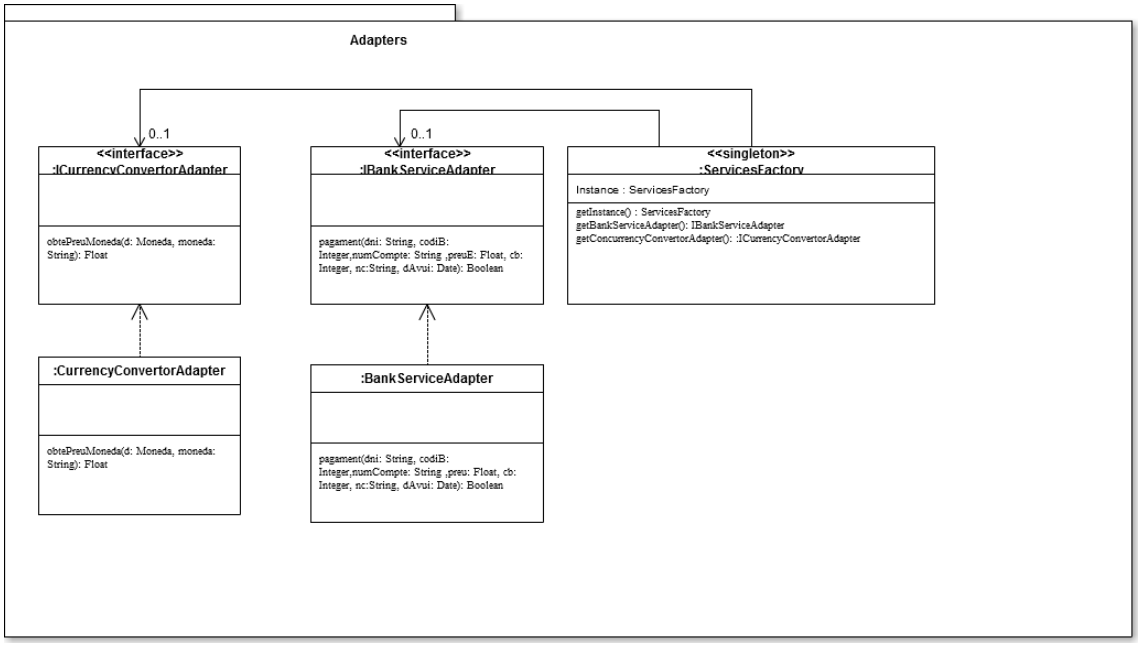

:Representació
:Espectacle
getInfo2()
- - - info = Ø
getTitol()
t
:Local
getNom()
n
:Sessió
getSessió()
s
etsEstrena?()
- - - false
b
- - - info.titol = t
- - - info.data = self.data
- - - info.nomLocal = n
- - - info.sessio = s
- - - info.nombreseientsIliures = self.nombreseientsIliures
- - - info.estrena = b
- - - info.preu = self.preu
info

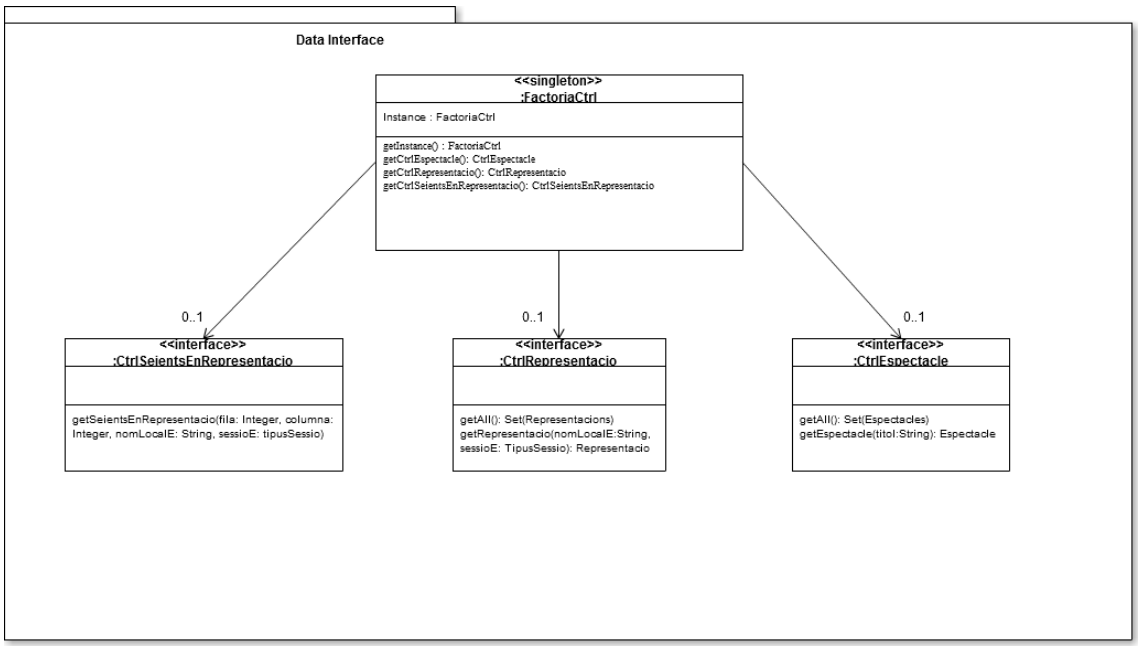
3.


Domain Model
Shows.com 1
codiBanc : Integer
númeroCompte : String
comissió : float
divisa : Moneda
canvis : Moneda[2]
getComissió() : Float
getCanvis() : Moneda[2]
getDivisa() : Moneda
getCodiBanc() : Integer
getNúmeroCompte() : String
<<enumeration>> Estat
ocupat
lliure
<<enumeration>> TipusSessió
matí
tarda
nit
<<enumeration>> Moneda
EUR
USD
GBP
Espectacle
títol : String
participants : Integer
consRep(data: Date) : Set(TupleType(nomLocal : String, sessió : String, nombreSeientsLliures : Integer, estrena : Boolean, preu : Float))
getTítol() : String
ERROR
Sessió
sessió : TipusSessió
getSessió() : TipusSessió
Local
nom : String
adreça : String
getNom() : String
Seient
fila : Integer
columna : Integer
getFila() : Integer
getColumna() : Integer
Representació
preu : float
data : date
nombreSeientsLliures : Integer
getData() : Date
getInfo() : Set(TupleType(nomLocal : String, sessió : String, nombreSeientsLliures : Integer, estrena : Boolean, preu : Float))
etsestrena() : Boolean
consOcup(nombEspectadors : Integer) : Set(TupleTyple(fila : Integer, columna : Integer))
getInfo2() : Set(TupleType(títol : String, data : Date, nomLocal : String, sessió : String, nombreSeientsLliures : Integer, estrena : Boolean, preu : Float))
getPreu() : Float
getRecàrrec() : Integer
associar(e : Entrada)
SeientEnRepresentació
estat : Estat
getEstat() : Estat
getInfo() : Set(TupleTyple(fila : Integer, columna : Integer))
associar(e : Entrada)
Entrada
identificador : String
dniClient : String
nombreEspectadors : Integer
data : date
- preu : float
crea(dni : String, nombEspectadors : Integer, data : Date, preu : Integer, rep : Representació, seients : Set(TupleTyple(fila : Integer, columna : Integer))
{dis|joint, complete}
Estrena
recàrrec : Integer
etsestrena() : Boolean
Faltan navegabilidades!


Domain Controllers
ControladorCasDusConsultarOcupacio
totesRepresentacions(): Set(TupleType(títol:string, data:Date, nomLocal:String, sessio:String, nombreSeientsLliures:Integer, estrena:Boolean, preu:Float))
consultaOcupacio(nomLocal:String, sessio:String, nombEspectadors:Integer): Set(seients:TupleType(fila:Integer, columna:Integer))
executar()
ControladorCasDusComprarEntrada
titolE: String
dataE: Date
nomLocal: String
sessioE: TipusSessio
nombEspectadorsE: Integer
seientsE: Set(TupleType(Fila:Integer, columna:Integer))
preuE: Float
obteEspectacles(): Set(títol:String)
obteRepresentacions(títol:String, data:Date): Set(TupleType(nomLocal:String, sessio:String, nombreSeientsLliures: Integer, estrena:Boolean, preu:Float))
obteOcupacio(nomLocal:String, sessio:String, nombEspectadors:Integer): Set(seients:TupleType(fila:Integer, columna:Integer))
seleccionarSeients(seients: Set(TupleType(fila:Integer, columna:Integer))): TupleType(preu:Float, canvis:Set(String))
obtePreuMoneda(moneda:String):Float
pagament(dni:String,codiB:Integer,numCompte:String)
executar()
ControladorCasDusConsultarRepresentacions
consultaEspectacles(): Set(títol:String)
consultaRepresentacions(títol:String, data:Date): Set(TupleType(nomLocal:String, sessio:String, nombreSeientsLliures: Integer, estrena: Boolean, preu:Float))
executar()
<<singleton>> :FactoriaCtrlCasosDus
Instance : FactoriaCtrlCasosDus
getInstance() : FactoriaCtrlCasosDus
getCtrlCasDusConsultaRepresentacions(): CtrlCasDusConsultaRepresentacions
getCtrlCasDusConsultaOcupacio():CtrlCasDusConsultaOcupacio
0..1
falta getCtrlCasdUsComprarEntrada...
Error: No debe haber Executar()


Service Locator
0..1
0..1
:ServeiCurrencyConvertor
conversionRate (d : Moneda, moneda : String) : Float
:BankService
autoritza(dni : String, codiB : Integer, numCompte : String, Import :Float, cb : Integer, nc : String, getDataActual()) : Boolean
<<singleton>>
:ServiceLocator
Instance : ServiceLocator
getInstance() : ServiceLocator
find(name :String) : ServiceReferece


Adapters
0..1
0..1
<<interface>>
:ICurrencyConvertorAdapter
obtePreuMoneda(d: Moneda, moneda: String): Float
<<interface>>
:IBankServiceAdapter
pagament(dni: String, codiB: Integer,numCompte: String ,preuE: Float, cb: Integer, nc:String, dAvui: Date): Boolean
<<singleton>>
:ServicesFactory
Instance : ServicesFactory
getInstance() : ServicesFactory
getBankServiceAdapter(): IBankServiceAdapter
getConcurrencyConvertorAdapter(): :ICurrencyConvertorAdapter
:CurrencyConvertorAdapter
obtePreuMoneda(d: Moneda, moneda: String): Float
:BankServiceAdapter
pagament(dni: String, codiB: Integer,numCompte: String ,preu: Float, cb: Integer, nc:String, dAvui: Date): Boolean


Data Interface
<<singleton>>
:FactoriaCtrl
Instance : FactoriaCtrl
getInstance() : FactoriaCtrl
getCtrlEspectacle(): CtrlEspectacle
getCtrlRepresentacio(): CtrlRepresentacio
getCtrlSeientsEnRepresentacio(): CtrlSeientsEnRepresentacio
0..1
0..1
0..1
<<interface>>
:CtrlSeientsEnRepresentacio
getSeientsEnRepresentacio(fila: Integer, columna: Integer, nomLocalE: String, sessioE: tipusSessio)
<<interface>>
:CtrlRepresentacio
getAll(): Set(Representacions)
getRepresentacio(nomLocalE:String, sessioE: TipusSessio): Representacio
<<interface>>
:CtrlEspectacle
getAll(): Set(Espectacles)
getEspectacle(titol:String): Espectacle

4.

Per emmagatzemar la informació que hem anat generant en els diagrames de seqüencia que necessitarem per altres operacions, hem utilitzat el patró Controlador Cas D'ús.

Hem fet servir el patró Creador a l'hora de decidir a quina classe assignar la responsabilitat de crear una instancia de la classe Entrada. En el nostre cas el creador serà el Controlador Cas D'ús Comprar Entrada.

El patró Factoria crea famílies d'objectes que estan relacionades i retorna la interfície que l'usuari sol·liciti, per tant, hem decidit utilitzar aquest patró a l'hora d'obtenir tots els controladors i adaptadors.

El Service Locator ens permet adquirir els serveis remots que necessitem.

Quan tenim classes amb una única instancia que han de ser accessibles des de diversos punts del sistema utilitzem el Singleton. Per això l'utilitzem en les factories, service locator i shows.com per que hem de poder accedir des de qualsevol cas d'ús.

Per últim, hem utilitzat el patró Adapter per convertir la funcionalitat d'obté preu moneda i de pagament en la funcionalitat que realment necessitem.