ISW – PRÀCTIQUES
LLIURAMENT 1

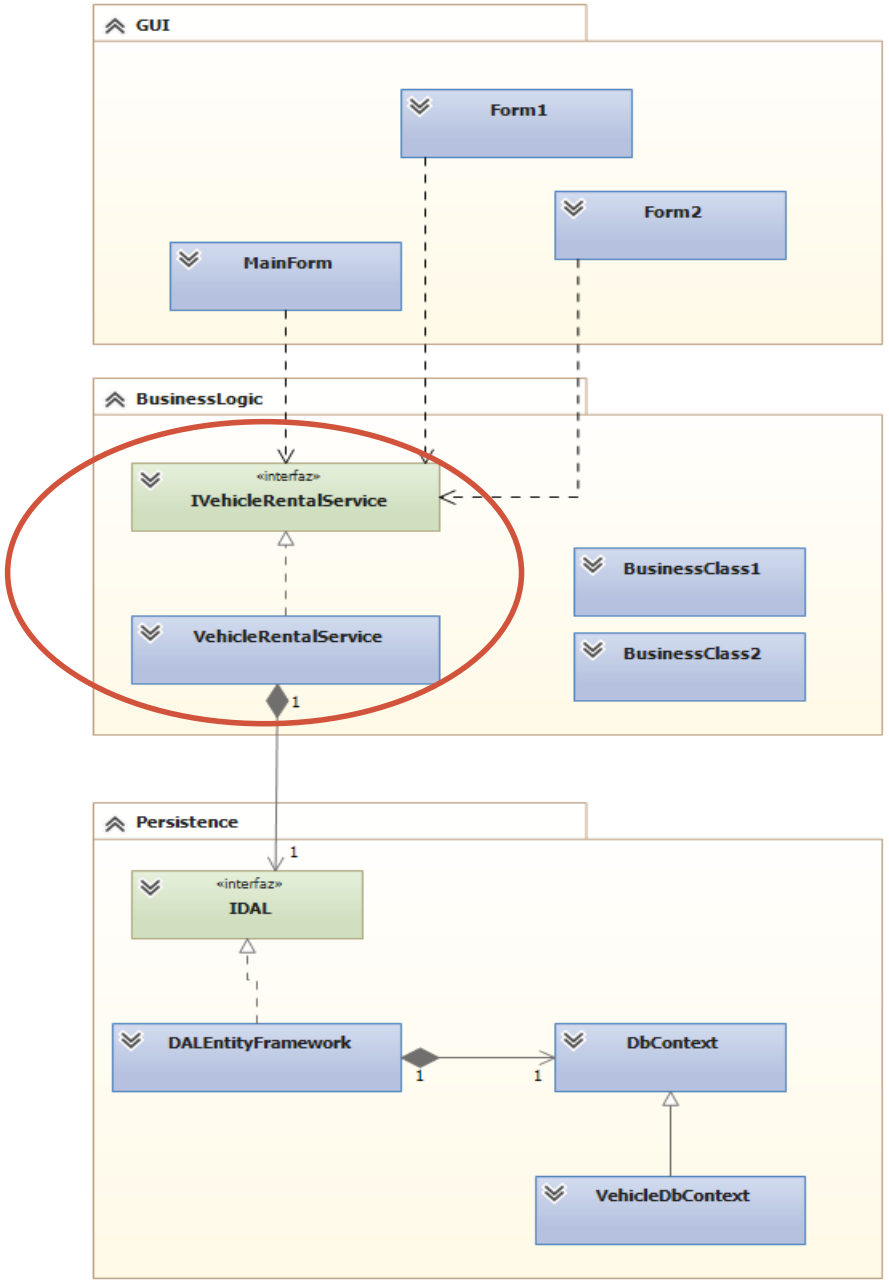
Casos d'ús– material d'ajuda

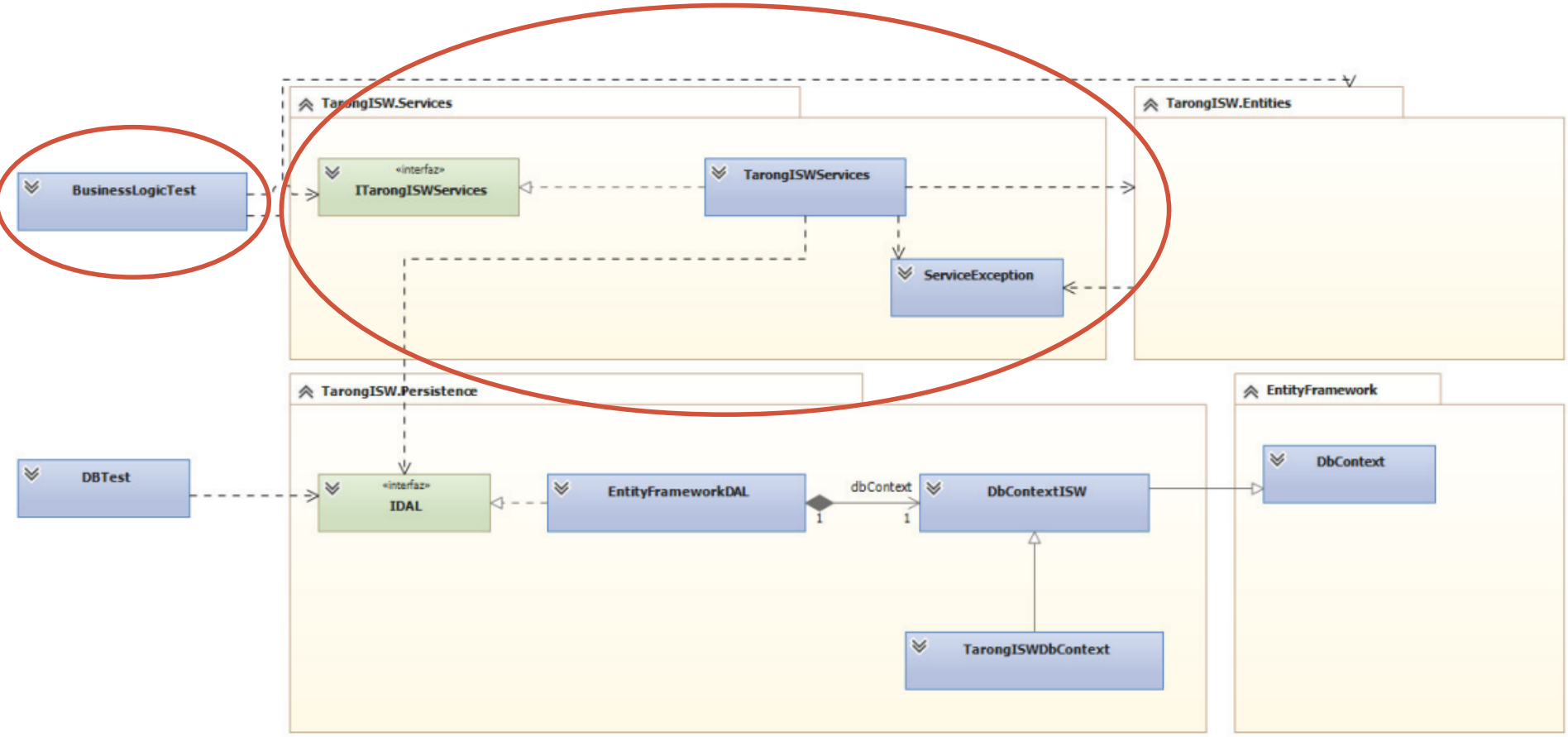
ETS Enginyeria Informàtica
DSIC – UPV

Curs 2021-2022

Disseny de la separació de capes

- 3 capes:
  - Presentació (IGU)
  - Lògica de Negoci
    - Classe que implementa els serveis de l'aplicació (casos d'ús).
  - Persistència




TarongISW.Services
TarongISW.Entities
BusinessLogicTest
«interfaz»
ITarongISWServices
TarongISWServices
ServiceException
TarongISW.Persistence
EntityFramework
DBTest
«interfaz»
IDAL
EntityFrameworkDAL
dbContext
1
1
DbContextISW
DbContext
TarongISWDbContext

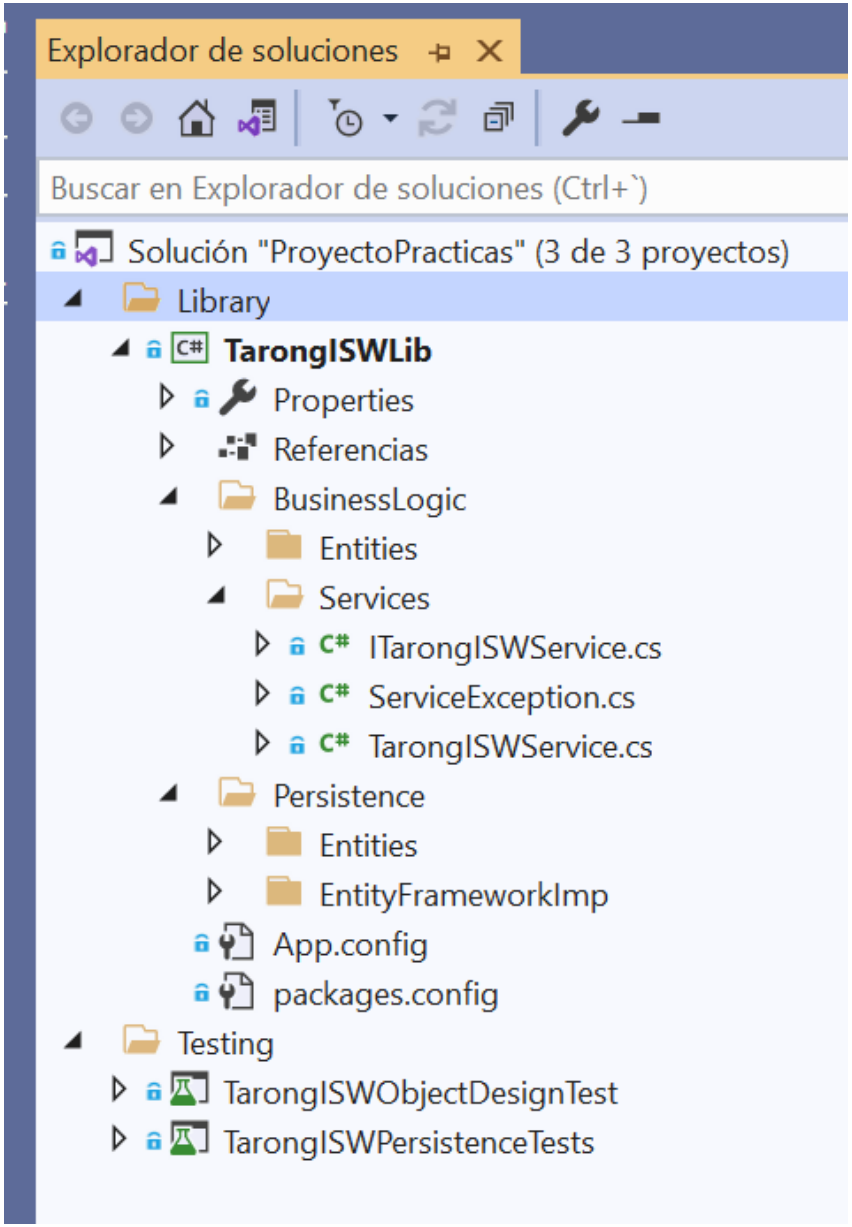
Capa Lògica

Nova carpeta ***Services*** a la carpeta ***BusinessLogic***

Aquesta carpeta contindrà:

- ***ITarongISWService.cs***
- ***ServiceException.cs***
- ***TarongISWService.cs***

** Part codi en PoliformaT



Capa Lògica

```
namespace TarongISW.Services
{
    1 referencia | fjaen, Hace 30 días | 1 autor, 1 cambio
    public class TarongISWService: ITarongISWService
    {
        private readonly IDAL dal;

        0 referencias | fjaen, Hace 30 días | 1 autor, 1 cambio
        public TarongISWService(IDAL dal)
        {
            this.dal = dal;
        }
```

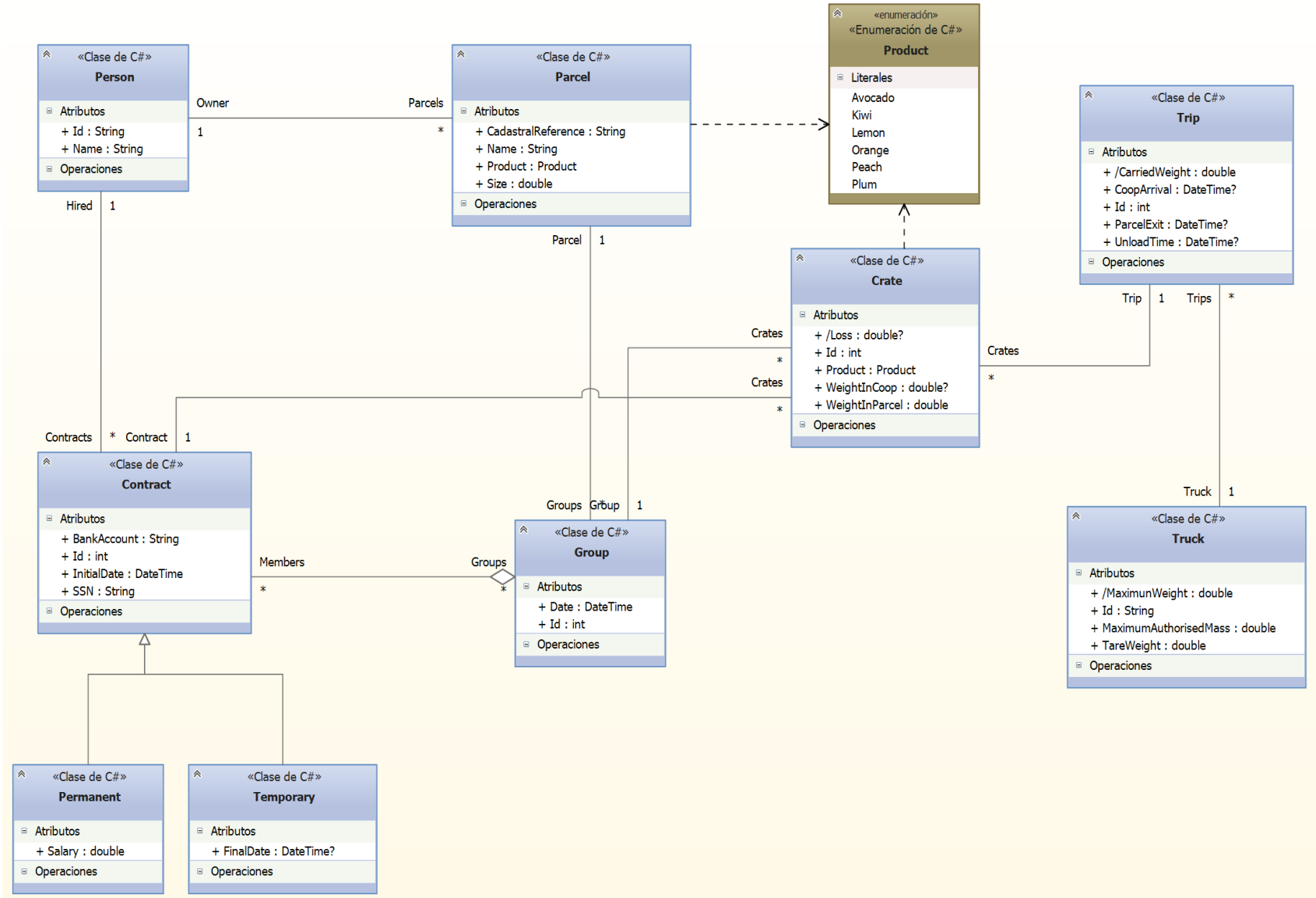
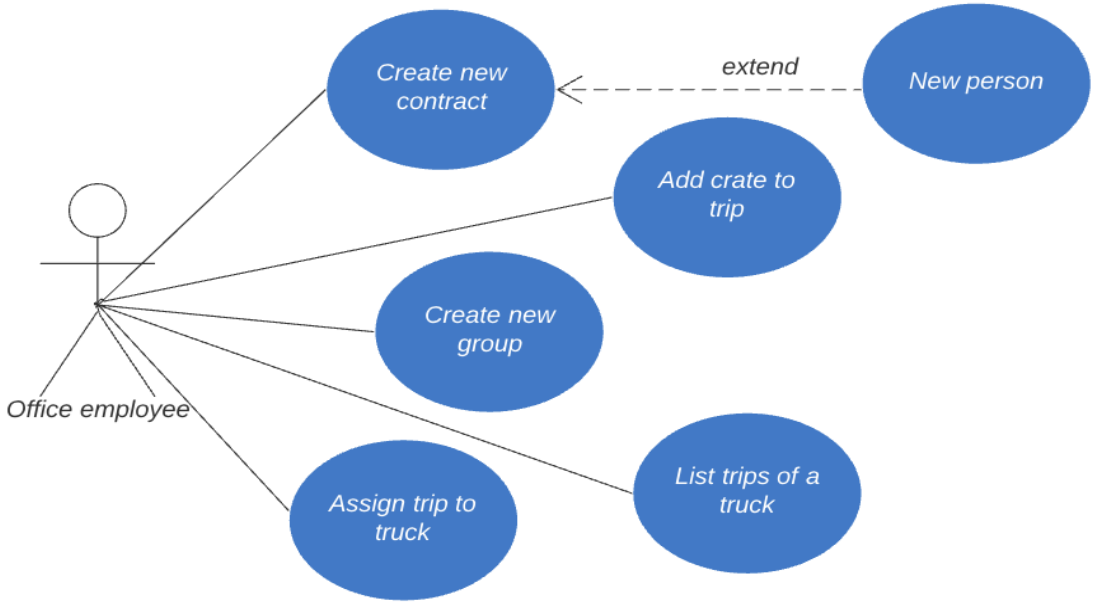

«Clase de C#»
Person
Atributos
+ Id : String
+ Name : String
Operaciones
Owner
1
Parcels
*
«Clase de C#»
Parcel
Atributos
+ CadastralReference : String
+ Name : String
+ Product : Product
+ Size : double
Operaciones
«enumeración»
«Enumeración de C#»
Product
Literales
Avocado
Kiwi
Lemon
Orange
Peach
Plum
«Clase de C#»
Trip
Atributos
+ /CarriedWeight : double
+ CoopArrival : DateTime?
+ Id : int
+ ParcelExit : DateTime?
+ UnloadTime : DateTime?
Operaciones
Hired
1
Parcel
1
«Clase de C#»
Crate
Atributos
+ /Loss : double?
+ Id : int
+ Product : Product
+ WeightInCoop : double?
+ WeightInParcel : double
Operaciones
Crates
*
Crates
*
Crates
*
Trip
1
Trips
*
Contracts
*
Contract
1
«Clase de C#»
Contract
Atributos
+ BankAccount : String
+ Id : int
+ InitialDate : DateTime
+ SSN : String
Operaciones
Groups
Group
1
«Clase de C#»
Group
Atributos
+ Date : DateTime
+ Id : int
Operaciones
Members
*
Groups
*
Truck
1
«Clase de C#»
Truck
Atributos
+ /MaximunWeight : double
+ Id : String
+ MaximumAuthorisedMass : double
+ TareWeight : double
Operaciones
«Clase de C#»
Permanent
Atributos
+ Salary : double
Operaciones
«Clase de C#»
Temporary
Atributos
+ FinalDate : DateTime?
Operaciones

Diagrama de casos d'ús



***Nota**: En la implementació considerem que només hi ha un actor*

Implementació de casos d'ús

- Identificar els serveis/mètodes necessaris per a implementar els casos d'ús.

ID	2
Use Case	**New person**
Actors	Office Employee
Goal	Create a new person
Description	1. The employee inserts the specific information of a person (dni, name) 2. The system creates the new person.
Precond	--
Postcond	The person is recorded
Synchronous Extension	If in 1, the user exists (dni), then an error message is shown and the employee may modify the data

- Consideracions:
  - Treballem amb un llenguatge orientat a objectes (reutilització i encapsulació).
  - Les operacions pròpies de les classes (per exemple, comprovar la igualtat) **han d'implementar-se a les classes, i no als serveis**. Així com accedir a les **col·leccions mitjançant els objectes de la capa de lògica** de negoci que mantenen aquestes col·leccions *(accedir mitjançant el DAL sols quan siga estrictament necessari)*

Capa Lògica

```
namespace TarongISW.Services
{
    public class TarongISWService: ITarongISWService
    {
        private readonly IDAL dal;

        public TarongISWService(IDAL dal)
        {
            this.dal = dal;
        }

        public void RemoveAllData()
        {
            dal.RemoveAllData();
        }

        public void Commit()
        {
            dal.Commit();
        }
```

Capa Lògica

```
namespace TarongISW.Services
{
    public class TarongISWService: ITarongISWService
    {
        . . .

        public void AddPerson(Person person)
                {
                    // Restricción: No puede haber dos personas con el mismo DNI
                    if (dal.GetById<Person>(person.Id) == null)
                    {
                        dal.Insert<Person>(person);
                        dal.Commit();
                    }
                    else throw new ServiceException("Person with Id " + person.Id + " already exists.");
                }

                public void AddParcel(Parcel parcel)
                {
                    // Restricción: No puede haber dos parcelas con el mismo nombre
                    if(!dal.GetWhere<Parcel>(x=>x.Name == parcel.Name).Any())
                    {
                        dal.Insert<Parcel>(parcel);
                        dal.Commit();
                    }
                    else throw new ServiceException("Parcel with Name " + parcel.Name + " already exists.");
                }
```

Capa Lògica

```
namespace TarongISW.Services
{
    public class TarongISWService: ITarongISWService
    {
        . . .

     public void AddTruck(Truck truck)
            {
                // Restricción: No puede haber dos camiones con la misma matrícula
                if (dal.GetById<Person>(truck.Id) == null)
                {
                    dal.Insert<Truck>(truck);
                    dal.Commit();
                }
                else throw new ServiceException("Truck with Id " + truck.Id + " already exists.");
            }
```

// incloure el vostre codi per a completar la capa de serveis...

```
        }
    }
```

Completant la capa de serveis...

ID	3
Use Case	**Create new group**[1]
Actors	Office Employee
Goal	The day of the harvesting of a parcel a group is created, with permanent and temporary workers who are assigned to the parcel.
Description	1. The employee requests to create a team 2. The system requests the parcel 3. The employee provides the corresponding parcel 4. The system requests the members of the team 5. The employee inserts or selects the employees who will be assigned to the group 6. The system creates the group and assigns it to the parcel
Precond	--
Postcond	The team is recorded
Synchronous Extension	In 3, the system verifies that there are no two groups the same day in the same parcel, otherwise it shows an error message. In 5, the system will not allow to add the same worker twice to the group. In 5, the system verifies that none of the members is assigned to another group the same day. If any member is already assigned the system shows an error message and go to 4 so that the employee may introduce the new data.

Localitzar una parcel·la (Parcel)

```
parcel=service.FindParcelById("123..");
```

Saber l'últim grup creat d'una parcel.la

```
g=parcel.LastGroup();
```

Buscar una persona

```
p=service.FindPersonById("xx");
```

Contracte actual d'una persona

```
C=p.LastActiveContract();
```

Assignar un treballador a un grup

```
g.AddMember(c);
```

Salvar el canvis

```
Service.Commit();
```

Ús de LINQ (Tema 6 – seminari 3)

https://www.campusmvp.es/recursos/post/introduccion-rapida-a-linq-con-c-sharp.aspx

ServiceException.cs

```
using System;

namespace TarongISW.Services

{
    public class ServiceException : Exception
    {
        public ServiceException()
        {
        }

        public ServiceException(string message)
        : base(message)
        {
        }

        public ServiceException(string message, Exception inner)
        : base(message, inner)
        {
        }
    }

}
```