

General description:

We need to modify an existing application that imports contacts from an MS Access database to our CRM Software. The data source (database and fields) are different in the new application.

Platform: **C#.NET 2005**

CRM Database: **MS SQL SERVER 2000**

Database to be imported: **MS Access 2000**

Web Application: **ASP.NET**

DLLs:

- App_Web_yr5sl3nj.dll
- App_Web_h2sqnpio.dll
- Castle.DynamicProxy.dll
- ClassLibrary_Usuarios_Netpoint.dll
- Crm.datos.dll
- Crm.datos.duplicados.dll
- Crm.negocio.dll
- HashCodeProvider.dll
- Iesi.Collections.dll
- log4net.dll
- Nhibernate.dll
- skmDataGrid.dll

Methods specification:

All companies stored in the Ms Access database that have to be imported to CRM will be shown in a ListBox control (sorted by "RazonSocial". In cases where "RazonSocial" is null or empty, they will be ordered by "NombreDeFantasía").

When the user select a company, all related data must be shown on screen.

To add a company to CRM must use all methods given by DLLs listed before.

Procedure for adding a company to CRM:

1. Using appropriate methods for searching potentially duplicated companies and/or repeated.

- **CRM.datos.duplicados.dll** has two filtering methods (*BuscarSimilares(Empresa)* and *BuscarDuplicados(Empresa)*). The difference between them is the filter level they do. One is more exact than the other in the search. Both methods use same input parameters and both return the same data type.

Input parameter: for both methods you have to pass Empresa entity as input parameter.

Output data: both methods return an ArrayList of similar or duplicated companies depending on used method.

Empresa entity: is defined by the class Empresa.cs and have all company data.

The difference between the data resulting from these two methods, is at the level of "similarity" possessing with respect to the company that we passed as input parameter.

The method BuscarSimilares (Empresa) returns companies that are possible to duplicate the companies that we have passed, these can have a match in the name or any other information. They have a level of "similarity" minor.

The method BuscarDuplicados (Empresa) returns companies that are duplicates almost certain that we have past, rarely duplicated these might fail, but the possibility exists. They have a level of "similarity" far superior to those returned by the previous method.

Two methods are applied to an object called **Buscador** which belongs to the same DLL.

The correct implementation of these two methods is performed automatically by calling in the first instance the method with a higher level of filtering, which generates two outcomes.

The first is obtained if one or more potential duplicate, in that the company will not charge the CRM, then we will see what to do with these duplicates.

The other is whether the returns as a result it has not found any potential duplicate, then call the method with a lower level of filtering.

This method, in turn, generate two other results. If this gets one or more similar potential, they added the company in the CRM, then we will see these as possible using similar. In the event that it fails to find any potential similar proceed to load the company in the CRM, then we will see how to do this.

2. If the company has no potential duplicates or repeated (ie it is a new company for CRM), add it using the methods.

Consider some of the things dll "hibernate".

To add or delete companies in the CRM must do so through sessions in "hibernate", as well as to collect data from companies. This can be seen in the source code for the application.

Once you have a company that we want to add to CRM with all its data, what we are going to do is save through:

```
long id = EmpresaACargar.Guardar(sessionHibernate);
```

Being sessionHibernate session of hibernate with the fact that we are managing the Empresa.

Getting this way id resulting from the company saved, which we will need to transform into what will be the IdCRM, as follows:

```
EmpresaACargar.Id = id;
if ((EmpresaACargar.IdCRM == null) || (EmpresaACargar.IdCRM
== string.Empty))
{
    for (long p = id; p < Int64.MaxValue; p++)
    {
        Empresa eT = new Empresa();
        eT.IdCRM = p.ToString();
        if (eT.Listar(session).Count == 0)
        {
            EmpresaACargar.IdCRM = p.ToString();
            break;
        }
    }
    EmpresaACargar.Guardar(session);
}
```

Now we have our company fully loaded in the CRM.

Finally we are going to make a clarification regarding the registration of persons within CRM, this procedure is done on an entirely separate procedure to add a company. The object Persona is an attribute belonging to entity Empresa. Once we have our entity Persona with their respective data, without forgetting that the company must be assigned through:

```
Persona.Empresa = EmpresaAAsignar;
```

Proceed to add, getting their id within CRM as follows:

```
long IdPersona = Persona.Guardar(sessionHibernate);
```

being sessionHibernate session of hibernate with the fact that we are managing Persona.

This concludes the process of adding a company CRM.

3. However, taking up a little explained in item 1. If the company has potential duplicates or similar (regardless which of the two methods we have been successful), will be required to display these results in a ListBox (sorted by its attribute "RazonSocial", if you do not have data in "RazonSocial", they will be sorted by "NombreDeFantasía"), where selecting any of the potential similar or duplicate, have to be displayed on-screen data which will compare with the data that is displayed on the screen of our company selected as to see if they are equal or not. The user must decide whether to add company CRM or not, then that will describe how the user can choose.

- 3.1. If the user checks the list of potential duplicates or similar, and concludes that there is none, should add the company to CRM using the methods described in step 2, but running the process through a button that the user can access with a name understandable for the user.
 ID_Ramo is similar to ID_RUBRO field in the project that we attach that we have to modify.
 In the access database we have a table called “Origen” with only one record. The containing of this record must be added to CRM to table “consulta” field “observaciones”. To fill “Fecha” field must use the date where the data is insterted and the user must be <NULL>. You have to use all methods as we used in the other project to insert data. This have to be inserted only in this case, that mean that is a “new” company for CRM. Not do that when the company exists.
- 3.2. If it is indicated wich of all potential similar or repeated is our company, you have to upgrade this company by completing the fields that are missing in CRM. Never should override those already loaded. In terms of Personas also have to be updated verifying if they are already loaded or not. If they are already loaded verify whether the e-mail is the same, in case that the email isn’t the same, add it to the data, and if the person ins’t in CRM, add it as we have seen in Step 2.

Some data has to be incorporated into the “company” (Empresa) and other to “people” (Persona) within the company.

RazonSocial ----- Company
 Domicilio ----- Company – only field domicilioFacturación, it has referente to another table that should only fill “calle” field. Other fields must be NULL
 Localidad ----- Company
 Provincia ----- Company
 Codigo Postal ----- Company
 Telefono ----- Company
 CUIT ----- Company
 Contacto ----- People
 Email ----- If it has the name of a contact People otherwise Company.
 Pagina Web ----- Company
 ID_Ramo-----Company

Of everything said in this paper, its implementation can be seen in the project attached to it, called EventContactsManager.sln.