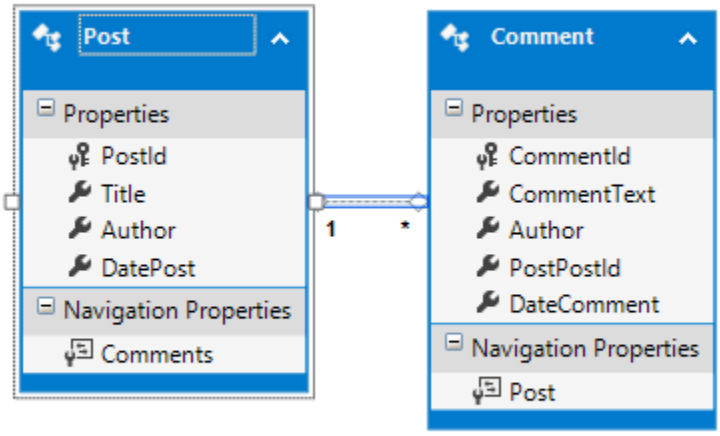
Aplicatie ASP.NET MVC ce foloseste un serviciu WCF

1. **Un proiect cu EF ce contine modelul (class library):**



In cadrul acestui proiect se defineste serviciul:

```
[ServiceContract]
public interface IServicePC
{
    [OperationContract]
    Post GetPostById(int id);

    [OperationContract]
    bool AMDPost(Post post); // AddModifyDeletePost

    [OperationContract]
    Post[] GetAllPosts();

    [OperationContract]
    Comment[] GetComments(Post post);

    [OperationContract]
    bool AMDComment(Post post, Comment comment);

   }
```

Clasele POCO vor constitui contractul de date, deci trebuiesc adnotate cu {DataContract] si [DataMember].

Aceste clase sunt derivate din BaseEntity:

```
public enum State
{
    Added,
    Unchanged,
    Modified,
```

```
    Deleted
}

[DataContract(IsReference = true)]
public abstract class BaseEntity
{
    protected BaseEntity()
    {
        State = State.Unchanged;
    }
    [DataMember]
    public State State { get; set; }
}

public static class EntityConvertState
{
    public static EntityState ConvertState(State state)
    {
        switch (state)
        {
            case State.Added:
                return EntityState.Added;
            case State.Modified:
                return EntityState.Modified;
            case State.Deleted:
                return EntityState.Deleted;
            default:
                return EntityState.Unchanged;
        }
    }
   }
```

si clasa Post (Comment este asemanatoare):

```
using System;
using System.Collections.Generic;
using System.Runtime.Serialization;
[DataContract(IsReference= true)]
public partial class Post:BaseEntity
{
    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage",
           "CA2214:DoNotCallOverridableMethodsInConstructors")]
    public Post()
    {
        this.Comments = new HashSet<Comment>();
    }

    [DataMember]
    public int PostId { get; set; }
    [DataMember]
    public string Title { get; set; }
    [DataMember]
    public string Author { get; set; }
    [DataMember]
    public System.DateTime DatePost { get; set; }

    [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage",
           "CA2227:CollectionPropertiesShouldBeReadOnly")]
```

```
        [DataMember]
        public virtual ICollection<Comment> Comments { get; set; }
       }
```

Implementarea interfetei IServicePC este urmatoarea:

```
public class ServicePC : IServicePC
    {
        public Post GetPostById(int id)
        {
            using (var context = new wcfEntities())
            {
                //context.Configuration.LazyLoadingEnabled = false;
                //context.Configuration.ProxyCreationEnabled = false;
                context.NoLazyLoading(); // contine ce e comentat mai sus
                var item = context.Posts.Where(x => x.PostId == id).FirstOrDefault();
                Console.WriteLine(item.ToString());
                return item;
            }
        }

        public bool AMDPost(Post post)
        {

              // trebuie implementata
            return true;
        }

        public Post[] GetAllPosts()
        {
            using (wcfEntities context = new wcfEntities())
            {
                context.NoLazyLoading();
                var items = context.Posts;
                Post[] posts = items.ToArray<Post>();
                return posts;

            }
        }

        public Comment[] GetComments(Post post)
        {
            // trebuie implementata
        }

        public bool AMDComment(Post post, Comment comment)
        {
           // cod
            return true;
        }
    }
```

Proiectul 2 este host-ul pentru serviciu

Este o aplicatie CUI

```
ServiceHost host = new ServiceHost(typeof(EFLayer.ServicePC),
          new Uri("http://localhost:8080/post"));
         //host.AddDefaultEndpoints();


         host.Open();
         var dd = host.Description;
         foreach (ServiceEndpoint sea in dd.Endpoints)
         {

             Console.WriteLine(" {0} {1} {2}", sea.Address, sea.Binding.Name,
            sea.Contract.Name);
         }
         Console.WriteLine("WCF started...");
             Console.ReadLine();
```

Proiectul 3 este aplicatia Web ASP.NET MVC ce foloseste acest serviciu.

Dupa ce s-a generat aplicatia se adauga referinta la serviciu.

Se lanseaza serviciul (proiectul 2).

In proiectul 3 se aduga referinta la serviciu, ***ServiceReference***, (nu mai folosim svcutil.exe).

Numele referintei este *ServiceReference1*.
Dupa ce am adaugat referinta la serviciu in web.config (la sfarsit) va trebui sa avem ceva de genul:

```
<system.serviceModel>
    <bindings>
      <basicHttpBinding>
        <binding name="BasicHttpBinding_IServicePC" />
      </basicHttpBinding>
    </bindings>
    <client>
      <endpoint address="http://localhost:8080/post"
                binding="basicHttpBinding"
          bindingConfiguration="BasicHttpBinding_IServicePC"
          contract="ServiceReference1.IServicePC"
          name="BasicHttpBinding_IServicePC" />
    </client>
</system.serviceModel>
```

Modificam actiunea Index() din controller Home astfel:

```
public ActionResult Index()
     {
           ServiceReference1.ServicePCClient pc =
                  new ServiceReference1.ServicePCClient();
         ServiceReference1.Post post = pc.GetPostById(1);
         return View(post);
      }
```

Aici parametrul e hard coded.
Modificam Index.cshtml

```
@model WebApplication1.ServiceReference1.Post
@{
    ViewBag.Title = "Home Page";
}

<div class="jumbotron">
    <h1>@Html.DisplayFor(model=>model.Title)</h1>
```

in rest fisierul ramane la fel.

Vrem sa demonstram ca ceea ce returneaza actiunea Index() apare in view Index.

Va ramane sa implementati celelalte actiuni si vizualizari.