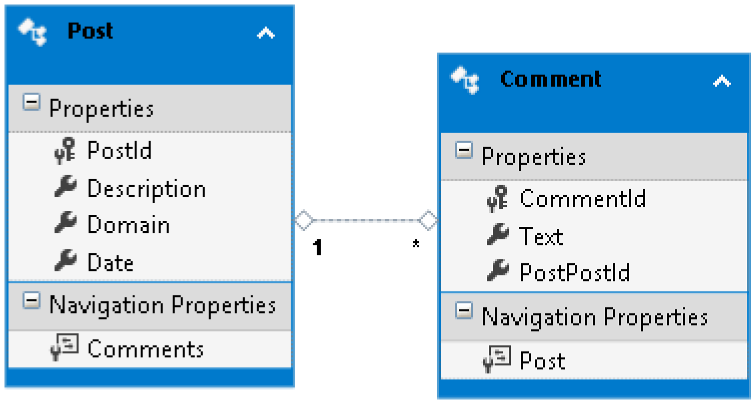
Laborator WCF – 24.04.2019

1. Problema prezentata la curs. Scop: urmarirea etapelor in realizarea proiectului.
2. WCF si EF – prezentat in continuare.

Etapa 1. Construire baza de date. Modelul ales este "Designer First". Proiect **ClassLibrary**.

Rezultatul este:



Clasele generate au fost modificate astfel:

```
//------------------------------------------------------------------------------
// <auto-generated>
//     This code was generated from a template.
//
//     Manual changes to this file may cause unexpected behavior in your application.
//     Manual changes to this file will be overwritten if the code is regenerated.
// </auto-generated>
//------------------------------------------------------------------------------

namespace PostComment
{
    using System;
    using System.Data.Entity;
    using System.Data.Entity.Infrastructure;

    internal partial class ModelPostCommentContainer : DbContext
    {
        public ModelPostCommentContainer()
            : base("name=ModelPostCommentContainer")
        {
            Configuration.LazyLoadingEnabled = false;
            Configuration.ProxyCreationEnabled = false;
        }
        protected override void OnModelCreating(DbModelBuilder modelBuilder)
        {
            throw new UnintentionalCodeFirstException();
        }

        public virtual DbSet<Post> Posts { get; set; }
        public virtual DbSet<Comment> Comments { get; set; }
    }
}
```

Clasa Post. Adnotare cu DataContract si DataMember.

```
//------------------------------------------------------------------------------
// <auto-generated>
//     This code was generated from a template.
//
//     Manual changes to this file may cause unexpected behavior in your application.
//     Manual changes to this file will be overwritten if the code is regenerated.
// </auto-generated>
//------------------------------------------------------------------------------

namespace PostComment
{
    using System;
    using System.Collections.Generic;
    using System.Runtime.Serialization;

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

        [DataMember]
        public int PostId { get; set; }
        [DataMember]
        public string Description { get; set; }
        [DataMember]
        public string Domain { get; set; }
        [DataMember]
        public System.DateTime Date { get; set; }

        [System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage",
               "CA2227:CollectionPropertiesShouldBeReadOnly")]
        [DataMember]
        public virtual ICollection<Comment> Comments { get; set; }
    }
}
```

Clasa Comment

```
//------------------------------------------------------------------------------
// <auto-generated>
//     This code was generated from a template.
//
//     Manual changes to this file may cause unexpected behavior in your application.
//     Manual changes to this file will be overwritten if the code is regenerated.
// </auto-generated>
//------------------------------------------------------------------------------

namespace PostComment
{
    using System;
    using System.Collections.Generic;
    using System.Runtime.Serialization;

    [DataContract(IsReference=true)]
    public partial class Comment
    {
        [DataMember]
        public int CommentId { get; set; }
        [DataMember]
        public string Text { get; set; }
        [DataMember]
        public int PostPostId { get; set; }
        [DataMember]
        public virtual Post Post { get; set; }
    }
}
```

Clasele Post si Comment au fost completate astfel:

```
using System.Collections.Generic;
using System.Linq;
using System.Data.Entity;

namespace PostComment
{
    public partial class Post
    {
        public bool AddPost()
        {
            using (ModelPostCommentContainer ctx = new ModelPostCommentContainer())
            {
                bool bResult = false;
                if (this.PostId == 0)
                {
                    var it = ctx.Entry<Post>(this).State = EntityState.Added;
                    ctx.SaveChanges();
                    bResult = true;
                }
                return bResult;
            }
        }
        public Post UpdatePost(Post newPost)
        {
            using (ModelPostCommentContainer ctx = new ModelPostCommentContainer())
            {
                Post oldPost = ctx.Posts.Find(newPost.PostId);
                if (oldPost == null) // nu exista in bd
                {
                    return null;
                }
                oldPost.Description = newPost.Description;
                oldPost.Domain = newPost.Domain;
                oldPost.Date = newPost.Date;
                ctx.SaveChanges();
                return oldPost;
            }
        }

        public int DeletePost(int id)
        {
            using (ModelPostCommentContainer ctx = new ModelPostCommentContainer())
            {
                return ctx.Database.ExecuteSqlCommand("Delete From Post where postid =
                      @p0", id);
            }
        }

        public Post GetPostById(int id)
        {
            using (ModelPostCommentContainer ctx = new ModelPostCommentContainer())
            {
                var items = from p in ctx.Posts where (p.PostId == id) select p;
                if (items != null)
                    return items.Include(c => c.Comments).SingleOrDefault();
                return null; // trebuie verificat in apelant
            }
        }

        public List<Post> GetAllPosts()
        {
            using (ModelPostCommentContainer ctx = new ModelPostCommentContainer())
            {
                return ctx.Posts.Include("Comments").ToList<Post>();
                // Obligatoriu de verificat in apelant rezultatul primit.
            }
        }

    }
}
```

Classa Comment

```
using System.Linq;
using System.Data.Entity;

namespace PostComment
{
    public partial class Comment
    {

        public bool AddComment()
        {
            using (ModelPostCommentContainer ctx = new ModelPostCommentContainer())
            {
                bool bResult = false;
                if (this == null || this.PostPostId == 0)
                    return bResult;
                if (this.CommentId == 0)
                {
                    ctx.Entry<Comment>(this).State = EntityState.Added;
                    Post p = ctx.Posts.Find(this.PostPostId);
                    ctx.Entry<Post>(p).State = EntityState.Unchanged;
                    ctx.SaveChanges();
                    bResult = true;
                }
                return bResult;
            }
        }

        public Comment UpdateComment(Comment newComment)
        {
            using (ModelPostCommentContainer ctx = new ModelPostCommentContainer())
            {
                Comment oldComment = ctx.Comments.Find(newComment.CommentId);
                // Deoarece parametrul este un Comment ar trebui verificata fiecare
                // proprietate din newComment daca are valoare atribuita si
                // daca valoarea este diferita de cea din bd.
                // Acest lucru il fac numai la modificarea asocierii.
                if (newComment.Text != null)
                    oldComment.Text = newComment.Text;
                if ((oldComment.PostPostId != newComment.PostPostId)
                            && (newComment.PostPostId != 0))
                {
                    oldComment.PostPostId = newComment.PostPostId;
                }
                ctx.SaveChanges();
                return oldComment;
            }
        }

        public Comment GetCommentById(int id)
        {
            using (ModelPostCommentContainer ctx = new ModelPostCommentContainer())
            {
                var items = from c in ctx.Comments where (c.CommentId == id) select c;
                return items.Include(p => p.Post).SingleOrDefault();
            }
        }
    }
}
```

Metodele de mai sus sunt metode ale instantei.

Varianta cu metode statice.

```
using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;

namespace PostComment.APIStatic
{
    public static class API
    {
        public static bool AddPost(Post post)
        {
            using (ModelPostCommentContainer ctx = new ModelPostCommentContainer())
            {
                bool bResult = false;
                if (post.PostId == 0)
                {
                    var it = ctx.Entry<Post>(post).State = EntityState.Added;
                    ctx.SaveChanges();
                    bResult = true;
                }
                return bResult;
            }
        }

        public static Post UpdatePost(Post newPost)
        {
            using (ModelPostCommentContainer ctx = new ModelPostCommentContainer())
            {
                // Ce e in bd. PK nu poate fi modificata
                Post oldPost = ctx.Posts.Find(newPost.PostId);
                if (oldPost == null) // nu exista in bd
                {
                    return null;
                }
                oldPost.Description = newPost.Description;
                oldPost.Domain = newPost.Domain;
                oldPost.Date = newPost.Date;
                ctx.SaveChanges();
                return oldPost;
            }
        }

        public static int DeletePost(int id)
        {
            using (ModelPostCommentContainer ctx = new ModelPostCommentContainer())
            {
```

```
            return ctx.Database.ExecuteSqlCommand("Delete From Post where postid =
                        @p0", id);
        }
    }

    /// <summary>
    /// Returnez un Post si toate Comment-urile asociate lui
    /// </summary>
    /// <param name="id"></param>
    /// <returns></returns>
    public static Post GetPostById(int id)
    {
        using (ModelPostCommentContainer ctx = new ModelPostCommentContainer())
        {
            var items = from p in ctx.Posts where (p.PostId == id) select p;
            if (items != null)
                return items.Include(c => c.Comments).SingleOrDefault();
            return null;
        }
    }

    /// <summary>
    /// Returnez toate Post-urile si Comment-urile corespunzatoare
    /// </summary>
    /// <returns></returns>
    public static List<Post> GetAllPosts()
    {
        using (ModelPostCommentContainer ctx = new ModelPostCommentContainer())
        {
            return ctx.Posts.Include("Comments").ToList<Post>();
        }
    }

    // Comment table
    public static bool AddComment(Comment comment)
    {
        using (ModelPostCommentContainer ctx = new ModelPostCommentContainer())
        {
            bool bResult = false;
            if (comment == null || comment.PostPostId == 0)
                return bResult;
            if (comment.CommentId == 0)
            {
                ctx.Entry<Comment>(comment).State = EntityState.Added;
                Post p = ctx.Posts.Find(comment.PostPostId);
                ctx.Entry<Post>(p).State = EntityState.Unchanged;
                ctx.SaveChanges();
                bResult = true;
            }
            return bResult;
        }
    }

    public static Comment UpdateComment(Comment newComment)
    {
        using (ModelPostCommentContainer ctx = new ModelPostCommentContainer())
        {
            Comment oldComment = ctx.Comments.Find(newComment.CommentId);
```

```
                if (newComment.Text != null)
                    oldComment.Text = newComment.Text;
                if ((oldComment.PostPostId != newComment.PostPostId)
                            && (newComment.PostPostId != 0))
                {
                    oldComment.PostPostId = newComment.PostPostId;
                }
                ctx.SaveChanges();
                return oldComment;
            }
        }

        public static Comment GetCommentById(int id)
        {
            using (ModelPostCommentContainer ctx = new ModelPostCommentContainer())
            {
                var items = from c in ctx.Comments where (c.CommentId == id) select c;
                return items.Include(p => p.Post).SingleOrDefault();
            }
        }
    }
}
```

Am terminat cu primul proiect.

Etapa 2. Proiect ClassLibrary. Obiectele din serviciu WCF.

Construire obiecte pentru WCF. Interfetele. Nu sunt luate in considerare metodele statice. Sunt comentate, partial.

```
using System.Collections.Generic;

using System.ServiceModel;
using PostComment;

namespace ObjectWCF
{
    [ServiceContract]
    interface InterfacePost
    {
        [OperationContract]
        bool AddPost(Post post);

        [OperationContract]
        Post UpdatePost(Post post);

        [OperationContract]
        int DeletePost(int id);

        [OperationContract]
        Post GetPostById(int id);

        [OperationContract]
        List<Post> GetPosts();
    }
    [ServiceContract]
    interface InterfaceComment
    {
        [OperationContract]
        bool AddComment(Comment comment);

        [OperationContract]
        Comment UpdateComment(Comment newComment);

        [OperationContract]
        Comment GetCommentById(int id);
    }

    [ServiceContract]
    interface IPostComment: InterfacePost, InterfaceComment
    {
    }
}
```

Implementare. Metodele statice nu sunt apelate. Sunt comentate.

```
using System;
using System.Collections.Generic;
using PostComment;
using PostComment.APIStatic;

namespace ObjectWCF
{
    public class PostComment : IPostComment
    {
        bool InterfaceComment.AddComment(Comment comment)
        {
            return comment.AddComment();
        }

        bool InterfacePost.AddPost(Post post)
        {
            return post.AddPost();
            //return API.AddPost(post);
        }

        int InterfacePost.DeletePost(int id)
        {
            Post post = new Post();
            return post.DeletePost(id);
            // static
            //return API.DeletePost(id);
        }

        Comment InterfaceComment.GetCommentById(int id)
        {
            Comment comment = new Comment();
            return comment.GetCommentById(id);
        }
        Post InterfacePost.GetPostById(int id)
        {
            // E nevoie de ac instanta. Metodele din API sunt metode ale instantei.
            Post post = new Post();
            // Mesaj ce apare in server CUI. Nu e necesar.
            Console.WriteLine("GetPostById. Id = {0}", id);
            post = post.GetPostById(id); // Neclar acest cod.
            Console.WriteLine("Post returnat. Id = {0} , Description = {1}",
                     post.PostId, post.Description);
            return post;
        }

        List<Post> InterfacePost.GetPosts()
        {
            Post post = new Post();
            return post.GetAllPosts();
        }

        Comment InterfaceComment.UpdateComment(Comment newComment)
        {
            return newComment.UpdateComment(newComment);
        }

        Post InterfacePost.UpdatePost(Post post)
        {
            return post.UpdatePost(post);
        }
    }
}
```

Etapa 3. Host pentru serviciu. App CUI.

```
using System;

using System.ServiceModel;
using ObjectWCF;
using System.ServiceModel.Description;

namespace HostWCF
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Lansare server WCF...");
            ServiceHost host = new ServiceHost(typeof(PostComment),
                     new Uri("http://localhost:8000/PC"));

            foreach (ServiceEndpoint se in host.Description.Endpoints)
                Console.WriteLine("A (address): {0} \nB (binding): {1}
                     \nC (Contract): {2}\n",
                    se.Address, se.Binding.Name, se.Contract.Name);

            host.Open();
            Console.WriteLine("Server in executie. Se asteapta conexiuni...");
            Console.WriteLine("Apasati Enter pentru a opri serverul!");
            Console.ReadKey();
            host.Close();
        }
    }
}
```

Fisier de configurare server

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <configSections>
    <!-- For more information on Entity Framework configuration, visit
http://go.microsoft.com/fwlink/?LinkID=237468 -->
  </configSections>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.6.1"/>
  </startup>

  <connectionStrings>
    <add name="ModelPostCommentContainer"
connectionString="metadata=res://*/ModelPostComment.csdl|res://*/ModelPostComment.ssdl|re
s://*/ModelPostComment.msl;provider=System.Data.SqlClient;provider connection
string=&quot;data source=IASIMIN-VOSTRO\SQL2012NEW;initial catalog=PostComment;integrated
security=True;MultipleActiveResultSets=True;App=EntityFramework&quot;"
providerName="System.Data.EntityClient" />
  </connectionStrings>
  <system.serviceModel>
    <services>
      <service name="ObjectWCF.PostComment" behaviorConfiguration="metadataSupport">
        <!--

        <endpoint address="http://localhost:8000/PC"
                  binding="basicHttpBinding"
                  contract="ObjectWCF.IPostComment"
                  name="BasicHttpBinding_IPostComment">
          <identity>
            <dns value="localhost"/>
          </identity>
        </endpoint>
        <endpoint           address="mex"
                            binding="mexHttpBinding"
                            contract="IMetadataExchange"
                            name="mexhttp"/>
      </service>
    </services>
    <behaviors>
      <serviceBehaviors>
        <behavior name="metadataSupport">
          <!-- Enables the IMetadataExchange endpoint in services that -->
          <!-- use "metadataSupport" in their behaviorConfiguration -->
          <!-- attribute. -->
          <!-- In addition, the httpGetEnabled and httpGetUrl -->
          <!-- attributes publish-->
          <!-- Service metadata for retrieval by HTTP/GET at the address -->
          <!-- "http://192.168.0.102:8000/SampleService?wsdl" -->
          <serviceMetadata httpGetEnabled="true" httpGetUrl=""/>
          <!-- <serviceMetadata/>-->
          <serviceDebug includeExceptionDetailInFaults="true"/>
        </behavior>
      </serviceBehaviors>
    </behaviors>

  </system.serviceModel>
</configuration>
```

Client. Un nou proiect Windows Forms.

Lanasare host.

Obtinere metadata cu svcutil. Adaugare fisiere generate la proiect.

Intr-o app Windows Forms codul arata astfel:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

using PostComment;

namespace ClientPostComment
{
    public partial class Form1 : Form
    {
        List<Post> posts = new List<Post>();

        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            posts = LoadPosts().ToList<Post>();
            dgp.DataSource = posts;
            dgp.Columns[0].Width = 0;
            if (dgp.Rows.Count > 0)
                dgc.DataSource = posts[0].Comments;
        }

        private static PostComment.Post[] LoadPosts()
        {
            PostCommentClient pc = new PostCommentClient();
            PostComment.Post[] p = pc.GetPosts();
            return p;
        }
        private void dgp_CellMouseClick(object sender, DataGridViewCellMouseEventArgs e)
        {
            if (e.RowIndex < 0)
                return;
            dgc.DataSource = null;
            dgc.DataSource = posts[e.RowIndex].Comments;
        }
    }
}
```

Formul principal contine doua DataGridView (dgp pentru Post si dgc pentru Comment). In rest descifrati codul.