Eötvös Loránd Tudományegyetem
Informatikai Kar

Webes alkalmazások fejlesztése

12. fejezet

Szolgáltatás alapú kommunikáció (WCF)

Giachetta Roberto

A jegyzet az ELTE Informatikai Karának 2016. évi jegyzetpályázatának támogatásával készült

Szolgáltatás alapú kommunikáció

Egységes kommunikációs platform

- A .NET keretrendszer számos technológiát kínál hálózaton kommunikáló, valamint elosztott alkalmazások fejlesztésére
- A *Windows Communication Foundation* (*WCF*) célja egy egységes felület megadása szolgáltatás alapú architektúra megvalósítására
  - ilyen befoglalt technológiák a webszolgáltatások (*WS*), a távoli eljáráshívás *(.NET Remoting*), üzenetfolyamok (*MSMQ*), stb.
  - az alapvető kommunikációs módszere a *SOAP*, de egyedi, XML alapú adatközlés is definiálható
  - hátránya, hogy a formátum miatt nagy adatmennyiséget generál

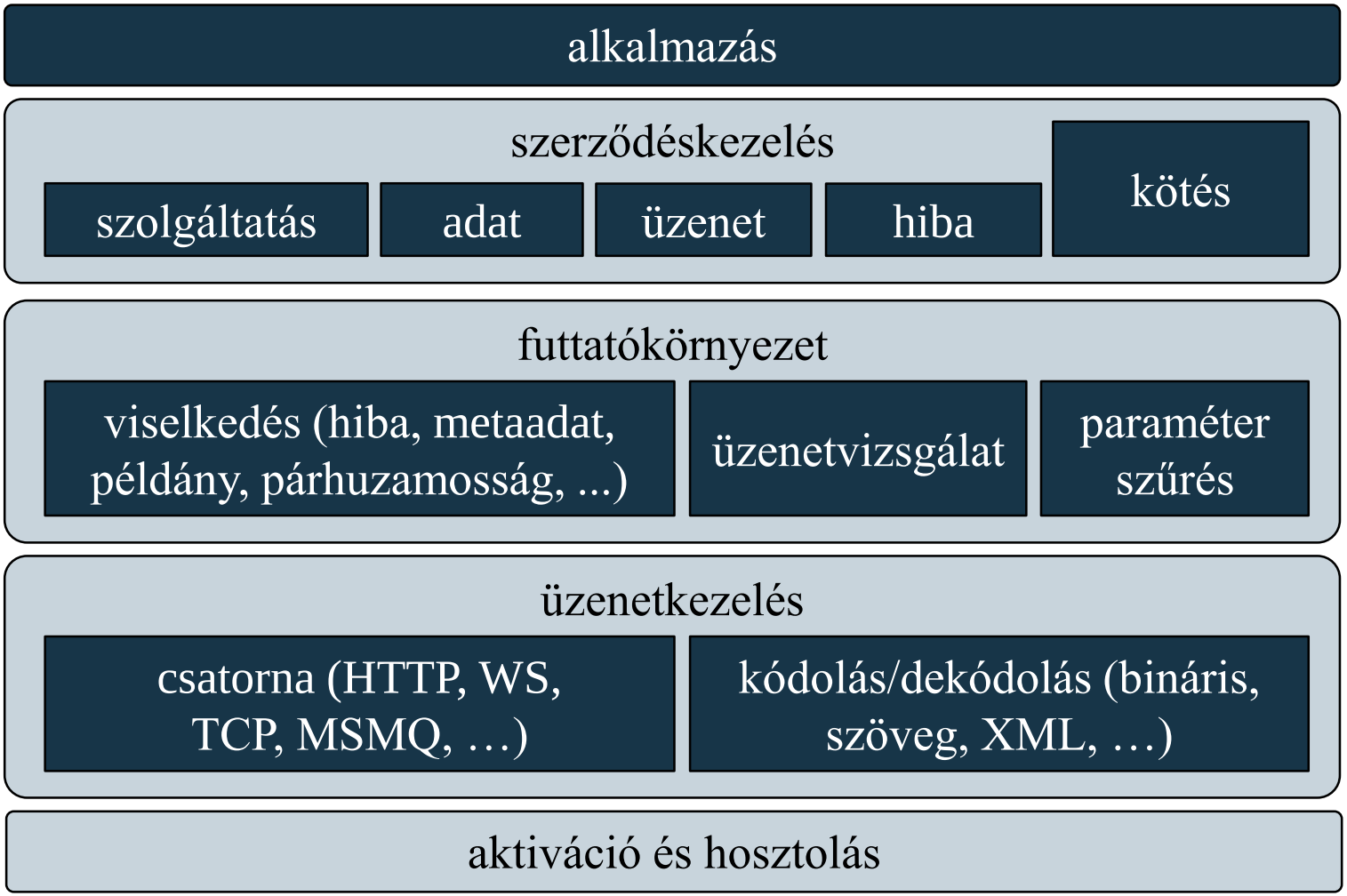
Szolgáltatás alapú kommunikáció

Egységes kommunikációs platform

- Előnyei:
  - a szolgáltatások lazán csatoltak, a kliensek kezelhetik függetlenül a szolgáltatásokat
  - könnyen beállítható, egyszerűen kezelhető
  - a webszolgáltatásnál nagyobb megbízhatóság és biztonság
  - kétirányúsítható a kliens-szerver csatorna
  - a WCF szolgáltatásokat biztosíthatja IIS, Windows Activation Service, Windows szolgáltatás, és lehet önhordozó
  - a folyamatok együttműködhetnek nem .NET-beli távoli eljáráshívókkal

Szolgáltatás alapú kommunikáció

Architektúra



Szolgáltatás alapú kommunikáció

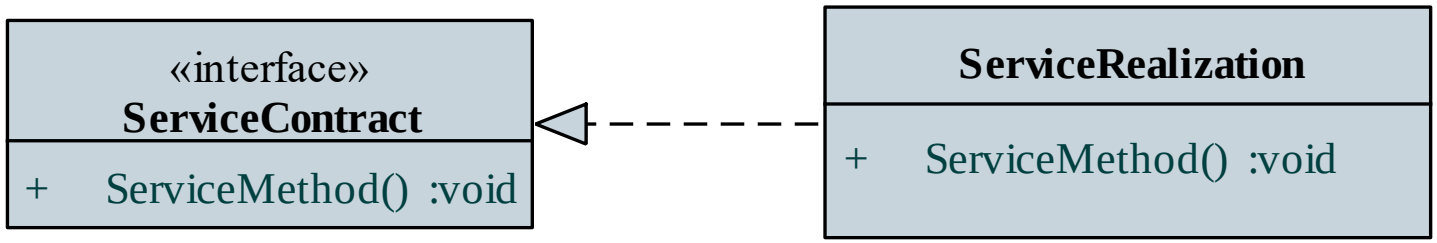
Kapcsolatok felépülése

- A WCF kapcsolatok alapjai a cím, a kötés és a szerződés (*Address/Binding/Contract*, *ABC*)
  - a *cím* a szolgáltatás elérési útvonala (URL), amely egyedileg azonosítja azt
  - a *kötés* megadja a kommunikáció módját, azaz a felhasználandó adatközlési csatornát (pl. SOAP over HTTP, SOAP over TCP), a protokollokat (pl. titkosításhoz), valamint a kódolást (pl. tömörítési mód)
  - a *szerződés* tartalmazza a szolgáltatott tartalmat, azaz mely műveletek, illetve adattípusok érhetőek el
    - a szerződést a keretrendszer a kód alapján automatikusan generálja futási időben

Szolgáltatás alapú kommunikáció

Szolgáltatás szerződés

- A legalapvetőbb szerződés a *szolgáltatás szerződés* (*Service Contract*), amely megadja, milyen funkciókat biztosít a szolgáltatás
  - a funkciók leképezhetőek függvényekre, pontosabban publikus metódusokra, amelyeket egy közös típusban helyezünk el
  - a szerződést interfészen keresztül adjuk meg, a szolgáltató osztály ezt az interfészt valósítja meg



Szolgáltatás alapú kommunikáció

Szolgáltatás szerződés

- az interfészt a **`ServiceContract`**, a műveleteket az **`OperationContract`** attribútumokkal jelöljük meg
  - itt csak a szolgáltatások szintaxisát adjuk meg, ami a szerződéshez szükséges
- pl.:

```
[ServiceContract] // szolgáltatás szerződés
public interface IEmployeeService
{
    [OperationContract]
    String GetEmployeeName(Int32 id);
    // szolgáltatás szintaxisa
    … // további szolgáltatások
}
```

Szolgáltatás alapú kommunikáció

Szolgáltatás szerződés

- a megvalósító osztálynál megadjuk a szolgáltatás viselkedési mintáját a **`ServiceBehavior`** attribútummal
- pl.:

```
[ServiceBehavior(Name="EmployeeService"
    ConcurrencyMode=ConcurrencyMode.Single)]
    // szolgáltatás viselkedési mintája
class EmployeeService : IEmployeeService
{ // szolgáltató osztály
    public String GetEmployeeName(Int32 id)
    {
        …
    } // szolgáltatás megvalósítása
    …
}
```

Szolgáltatás alapú kommunikáció

Szolgáltatás szerződés

- a szerződés viselkedése több módon konfigurálható, pl.:
  - név (**`Name`**) és névtér (**`Namespace`**)
  - párhuzamosság szabályozása (**`ConcurrencyMode`**, **`InstanceContextMode`**)
  - időkorlátok (**`SendTimeout`**, **`SendTimeout`**, …)
- ezen felül a műveletek viselkedése külön szabályozható az **`OperationBehavior`** attribútummal, pl.:
  - paramétermegsemmisítés (**`AutoDisposeParameters`**)
  - objektum létrehozási mód (**`ReleaseInstanceMode`**)
  - tranzakció-kezelés módja (**`TransactionAutoComplete`**)

Szolgáltatás alapú kommunikáció

Szolgáltatások futtatása

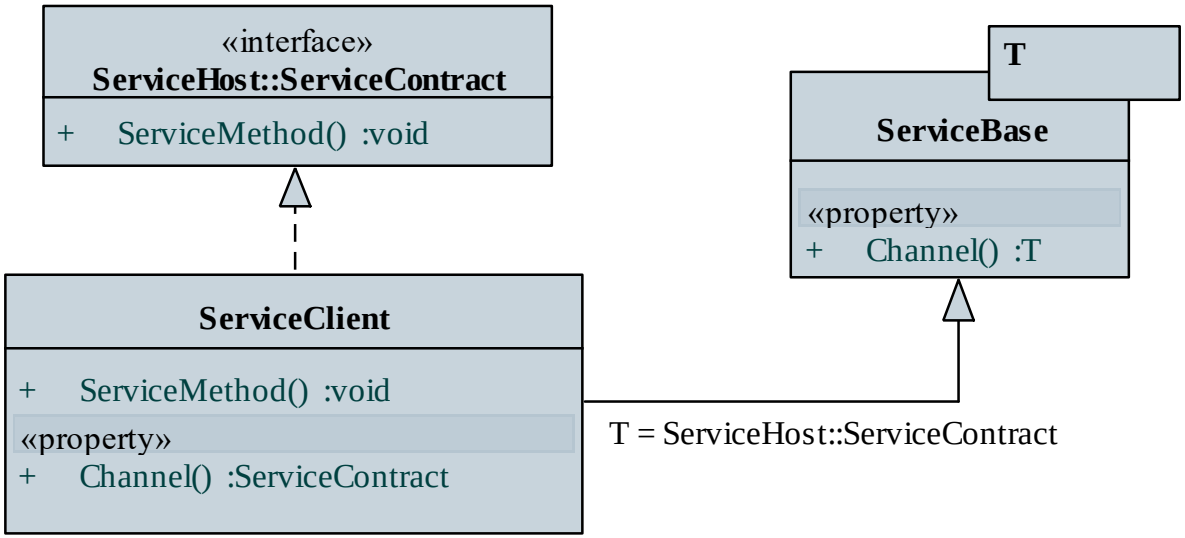
- A szolgáltatás futtatásához fel kell építeni egy szolgáltatót (*Service Host*), amit bármilyen alkalmazás létesíthet egy **`ServiceHost`** objektum segítségével
  - pl.:

```
Uri uri = new Uri("http://localhost/Service");
    // szolgáltatás címe
ServiceHost host = new
    ServiceHost(typeof(EmployeeService), uri);
    // hoszt létrehozása
host.AddServiceEndpoint(
    typeof(IEmployeeService),
    new WSHttpBinding(), uri);
    // szolgáltatás végpontjának megadása
```

Szolgáltatás alapú kommunikáció

Szolgáltatások felhasználása

- A kliens oldalon egy megfelelő kliens osztály fogadja a kapcsolatot, amely a **`ClientBase<T>`** osztály leszármazottja, és emellett megvalósítja a szerződés interfészét
  - a végrehajtást a **`ClientBase`** osztály **`Channel`** tulajdonságán keresztül érhetjük el (szintén megvalósítja az interfészt)



Szolgáltatás alapú kommunikáció

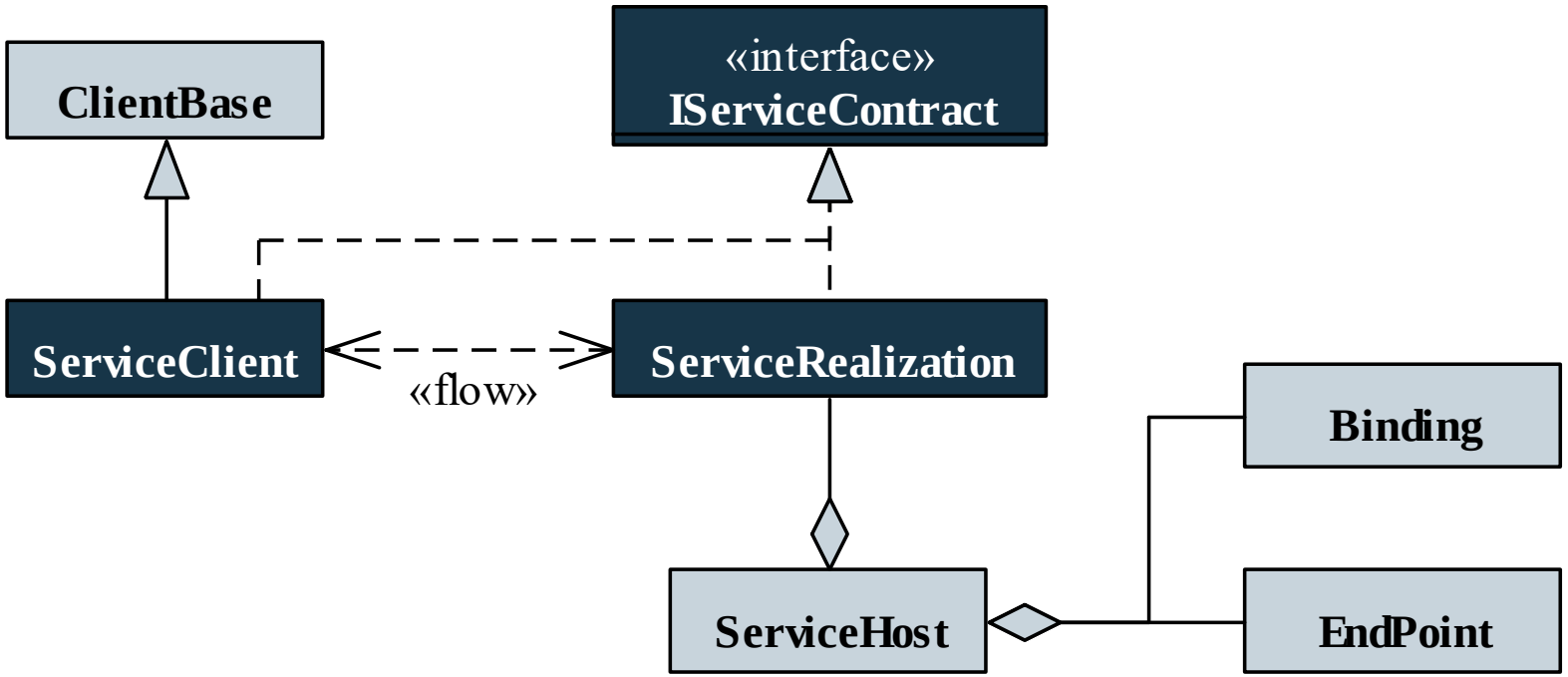
Szolgáltatások felhasználása

- pl.:

```
public partial class HelloWorldClient :
      ClientBase<IEmployeeService>,
      IEmployeeService
   // a szolgáltatás kliens, amely megvalósítja
   // a szerződést, illetve a ClientBase
   // osztályt
{
   public String GetEmployeeName(Int32 id)
   {
      return client.Channel.GetEmployeeName(id);
      // a Channel tulajdonságon át hívhatunk
   }
}
```

Szolgáltatás alapú kommunikáció

Szolgáltatások megvalósítása



Szolgáltatás alapú kommunikáció

Példa

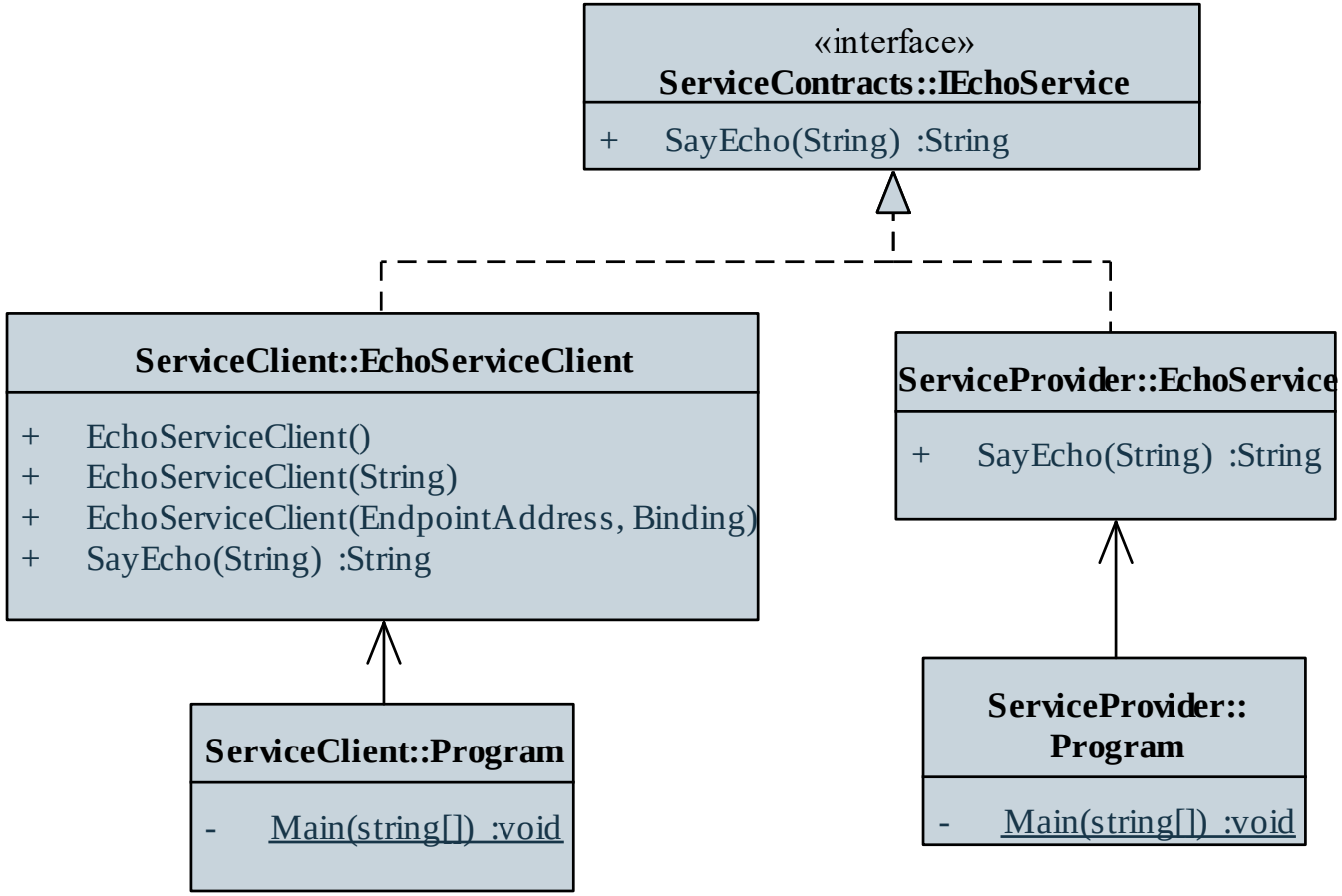
Feladat: Készítsünk egyszerű WCF szolgáltatást, valamint klienst, amely egy visszhangot biztosít, azaz visszaküldi a tartalmat.

- legyen a szerződés neve **`IEchoService`**, amelynek egyetlen művelete a **`SayEcho`**, mely szöveget kap paraméterben és szöveget ad vissza
- vegyünk fel három projektet, egyik tartalmazza a szerződést (**`ServiceContracts`**), egyik a szolgáltatót (**`ServiceProvider`**), egyik pedig a klienst (**`ServiceClient`**)
- a szolgáltató és a kliens is egyszerű konzolos alkalmazások lesznek

Szolgáltatás alapú kommunikáció

Példa

Tervezés:



Szolgáltatás alapú kommunikáció

Példa

Megvalósítás (`IEchoService.cs`):

```
[ServiceContract]
    // szolgáltatás szerződés
public interface IHelloWorldService
{
    [OperationContract] // művelet
    String SayHello(String name);
}
```

Szolgáltatás alapú kommunikáció

Példa

Megvalósítás (`Program.cs`):

```
…
public class EchoService :
        IEchoService {
    // a szerződést megvalósító szolgáltatás
    public String SayEcho(String name){
        return String.Format(…, name);
    }
}
…
using (ServiceHost host =
    new ServiceHost(typeof(EchoService),
        baseAddress)) {
        // létrehozzuk a szolgáltatás hosztját
```

Szolgáltatás alapú kommunikáció

Példa

Megvalósítás (`Program.cs`):

```
// kapcsolati protokoll (kötés) létrehozása
BasicHttpBinding binding =
   new BasicHttpBinding();
…
// végpont felvétele
host.AddServiceEndpoint(
   typeof(IEchoService), binding, "");
ServiceMetadataBehavior smb =
   new ServiceMetadataBehavior();
…
host.Description.Behaviors.Add(smb);
host.Open(); // hoszt indítása
…
```

Szolgáltatás alapú kommunikáció

Szolgáltatások automatikus létesítése

- A szolgáltatás kódban történő megadása sokszor körülményes, de lehetőségünk van a hosztolást automatikusan generálni
  - a szolgáltatást hosztolhatja IIS webszerver (*WCF Service Application*), vagy lehet önhosztoló (*WCF Service Library*)
  - mindkét esetben csak a szolgáltatást (a szerződést és megvalósítását) kell megadnunk, illetve egy konfigurációt (**`web.config`**, vagy **`app.config`**), amely a szolgáltatás beállításait tartalmazza
    - a szolgáltató objektumokat, és azok paraméterezését a keretrendszer a konfiguráció alapján generálja
  - lehetőségünk van *Windows Activation Service* segítségével futtatni a szolgáltatást, amely további funkciókat nyújt

Szolgáltatás alapú kommunikáció

Szolgáltatások automatikus létesítése

- Amennyiben a szerver szolgáltat metaadatokat, lehetőségünk van a kliensben szükséges osztályok automatikus generálásához (*Add Service Reference*…)
  - az így létrehozott névtérben megkapjuk a klienskezelőt (`ClientBase` leszármazott), valamint az adatszerződéseket
  - a hívások szinkron és aszinkron (`async`) formában is generálódnak
  - a beállítások kliens oldalon is a konfigurációs fájlban tárolódnak (`app.config`)
- Ha a kliens nem éri el a szolgáltatást, nem egyezik a szerződés, a kliens objektum a műveletre megfelelő kivételt vált ki

Szolgáltatás alapú kommunikáció

Példa

Feladat: Készítsünk egyszerű WCF szolgáltatást, valamint klienst, amely egy visszhangot biztosít, azaz visszaküldi a tartalmat.

- legyen a szerződés neve **`IEchoService`**, amelynek egyetlen művelete a **`SayEcho`**, mely szöveget kap paraméterben és szöveget ad vissza
- generáljuk automatikusan a hoszt és kliens osztályokat, konfiguráció alapján, az alapértelmezett konfiguráció megfelelő lesz
- így már csak két projektre lesz szükségünk, egyik a szolgáltató (*WCF Service Application*), másik a kliens (konzol alkalmazás)

Szolgáltatás alapú kommunikáció

Szolgáltatások automatikus létesítése

- Pl. (`app.config`):

```
…
<system.serviceModel>
    <bindings>
        <basicHttpBinding>…</basicHttpBinding>
    </bindings>
    <client>
        <endpoint address="…"
            binding="basicHttpBinding"
            bindingConfiguration="…"
        contract="EchoServiceReference.IEchoService"
            name="BasicHttpBinding_IEchoService" />
    </client>
</system.serviceModel>
```