Klientrakenduse ehitamine

Generaatori kasutamine

Selleks et päringuid xtee teenuste poole teha, tuleb kõigepealt valmis genereerida wsdl-ist struktuurid. Avades Xtee Generaatori, tuleb sisestada dto-de ja serialiseerijate väljundite ning wsdli asukohad. Wsdli asukoht võib olla nii veebiaadress kui ka asukoht failisüsteemis, mõlemaga saab generaator hakkama. Samuti ei pea olema erinev dto-de ja serialiseerijate genereerimise asukoht, see võib olla ühesugune.
Kui genereerimine on õnnestunud, siis tuleb väljundist luua klass teegi projekt ja ära kompileerida (kui dtod ja serialiseerijad olid erinevatesse kataloogidesse paigutatud siis 2 projekti).
Vahemärkuseks, et generaatoril on konfiguratsiooni fail (Seadistus.xml), kuhu saab kirjeldada geneerimiseks argumendid ning mida saab valida vahendil olevast rippmenüüst. Näiteks nii:

```
<keha xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <item xsi:type="tns:XteeGenConfiguraton"
xmlns:tns="http://XteeGen/XteeGenConfiguraton.xsd">
    <Name xsi:type="xsd:string">Wsdl1</Name>
    <WsdlPath xsi:type="xsd:string">http://local/app.wsdl</WsdlPath>
    <ModelOutputFolder xsi:type="xsd:string">C:\app1</ModelOutputFolder>
    <ImplOutputFolder xsi:type="xsd:string">C:\app1</ImplOutputFolder>
    <GenerateOperationsNamely xsi:type="xsd:boolean">true</GenerateOperationsNamely>
  </item>
 <item xsi:type="tns:XteeGenConfiguraton"
xmlns:tns="http://XteeGen/XteeGenConfiguraton.xsd">
    <Name xsi:type="xsd:string">Wsdl2</Name>
    <WsdlPath xsi:type="xsd:string">c:\\app2.wsdl</WsdlPath>
    <ModelOutputFolder xsi:type="xsd:string">C:\app2</ModelOutputFolder>
    <ImplOutputFolder xsi:type="xsd:string">C:\app2</ImplOutputFolder>
    <GenerateOperationsNamely xsi:type="xsd:boolean">true</GenerateOperationsNamely>
  </item>
</keha>
```

GenerateOperationsNamely element konfiguratsioonis on igand vanemast versioonist kus meetodite signatuurid genereeriti kõik ühe nimega (Execute) mis aga tekitas probleeme sellega

et operatsioonide hulgas võib olla mitu sama sisendi/väljundiga operatsioone ja geneeruv kood seega ei

komileeru. Soovitaks alati selle elemendi väärtuse tõesena hoida, siis klientrakendusele genereeruva teenuse kliendi meetodid nimetatakse wsdlis olevate operatsioonide nime järgi ja nii ei tohiks konflikte tekkida.

Klientrakenduse ülesseadmine

Xtee päringu väljakutsumiseks on vaja lisada rakendusse genereeritud dll-id ja Xtee.Core.dll (Lisaks log4net.dll).
Seejärel saab juba kirjutada päringu, mis võib välja näha näiteks selline:

```
namespace Client
{
    class Program
    {
        static void Main(string[] args)
        {
            //CalcAdapter on pärit genereeritud struktuuride teegist
            var client = new CalcAdapter();
            decimal result = client.Add(new Variables(3, 4)).Value;
            result = client.Multiply(new Variables(3, 4)).Value;
            result = client.Subtract(new Variables(3, 4)).Value;
            Console.WriteLine(result);
        }
    }

}
```

Selleks et päringud reaalselt ka tööle hakkaksid, on vaja rakendus konfigureerida.
Konfiguratsiooni failis tuleb registreerida xtee sektsioon selliselt:

```
  <configSections>
    <section name="xtee.configuration"
type="Xtee.Core.Client.Config.ClientConfigurationSection, Xtee.Core"/>
  </configSections>
```

ning kliendi konfiguratsioon minimaalses mahus selliselt:

```
<xtee.configuration proxyURL="http://localhost/adapter.asmx">
    <xteeTypeAssemblies>
      <clear/>
      <add assemblyName="GenereeritudDtoDllNimi"/>
      <add assemblyName="GenereeritudSerialiseerijadDllNimi"/>
    </xteeTypeAssemblies>
  </xtee.configuration>
```

Kindlasti tuleb registreerida genereeritud dll-id, nagu näites, muidu päringud tööle ei hakka.

Xtee konfiguratsiooni võimalikud atribuudid on:

atribuut	**väärtus**
proxyURL	Teenuse url mille vastu päringud sooritatakse
storeMessages	(true/false) Kas sõnumid salvestada või mitte
storagePath	asukoht kuhu sõnumid salvestada, võib olla nii relatiivne kui absoluutne rakenduse suhtest
timeout	saab täpsustada kaua oodatakse enne kui päring katkestatakse, ooteaeg on millisekundites
isikukood	xtee parameeter, saab vaikimisi määrata isikukoodi, mis päises kaasa antakse
asutus	xtee parameeter, saab vaikimisi määrata asutuse, mis päises kaasa antakse
toimik	xtee parameeter, saab vaikimisi määrata toimiku, mis päises kaasa antakse

Konfiguratsiooni saab hõlpsasti muuta ka koodi kaudu, tavaliselt võib olla näiteks isikukood muutuv vastaval kasutajale:

```
static void Main(string[] args)
        {
            var client = new CalcAdapter();
            client.XteeCommand.Configuration.Isikukood = "12345678912";
        }
```

Serverrakenduse ehitamine

Generaatori kasutamine

Esmalt on vaja genereerida struktuurid ja kompileerida teeki samamoodi nagu eespool kirjeldatud.

Serverrakenduse ülesseadmine

Klientide teenindamiseks tuleb koostada teenuse klass, panna pärinema soovitud teenuse interface-ist, mis on kättesaadav geneeritud teegist (geneeritakse wsdlist operatsioonidele vastavad interface-id) ja implementeerida ProcessRequest meetod:

```
namespace Server.Services
{
    public class ServiceMultiply : Xtee.XteeAdapter.Calc.MultiplyServiceHandler
    {
        public MultiplyResponse ProcessRequest(MultiplyRequest input, StandardHeader
header)
        {
            return new MultiplyResponse(new Result(input.Keha.VarX *
input.Keha.VarY));
        }

    }
}
```

Selleks et teenus ka tööle hakkaks, tuleb konfigureerida rakendus:

Registreerida sektsioon konfiguratsiooni failis:

```
<configSections>
    <section name="adapter.configuration"
type="Xtee.Core.Adapter.Service.Config.AdapterServiceConfigurationSection,
Xtee.Core"/>
  </configSections>
```

Ja konfiguratsioon ise:

```
<adapter.configuration storagePath="stored_messages" storeMessages="true">
    <services>
      <clear/>
      <!--Xtee operatsioonid seotakse konkreetse klassiga mis teenendab konkreetset
xtee
            operatsiooniga   -->
      <add producer="Calc" name="Add" version="v1"
servicehandler="Server.Services.ServiceAdd,Server"/>
      <add producer="Calc" name="Multiply" version="v1"
servicehandler="Server.Services.ServiceMultiply,Server"/>
      <add producer="Calc" name="Subtract" version="v1"
servicehandler="Server.Services.ServiceSubtract,Server"/>
    </services>
    <xteeTypeAssemblies>
      <clear/>
        <add assemblyName="GenereeritudDtoDllNimi"/>
      <add assemblyName="GenereeritudSerialiseerijadDllNimi"/>
    </xteeTypeAssemblies>
  </adapter.configuration>
```

Lisaks tuleb regisreerida teenuse httphandler:

```
 <system.web>
    <httpHandlers>
      <add verb="*" path="AdapterXtee.asmx"
type="Xtee.Core.Adapter.XteeServiceHandlerFactory"/>
    </httpHandlers>
    .........
 </system.web>
```

Httphandleri-i registreerimine IIS7-es käib natukene teistmoodi, näites tooduna töötab IIS6-es ja ASP.NET development serveris .

Failide saatmisest

Antud lahendus võimaldab saata ja vastuvõtta faile MIME formaadis. Lähtekoodis on ka näiterakendus failide saatmisest ja vastuvõtmisest: Examples\DocumentRepository.

Kuna Soap spetsifikatsioon pole päris piisav, et saaks genereerida sellised struktuurid, mis käsitleks faile

mugaval viisil, siis peab wsdl-i lisama paar nüanssi, et saada failid dto-de küljest kätte ilma ülearuse peavaluta. Selleks et dto-d ja serialiseerijad oskaks faile käsitleda, tuleb wsdl-is deklareerida atribuut mimeAttachment ja faili käsitleva objekti property atribuudiga tuleb öelda, et see käsitleb Mime faili:

```
<attribute name="mimeAttachment" type="boolean"/>
<complexType name="Document">
  <all>
    <element name="Name" type="string"></element>
      doc prefix mimeAttachmendi ees on TargetNamespace-i prefix, ehk siis lokaalse nimeruumi prefix
    <element name="Sisu" nillable="true" doc:mimeAttachment="true" type="SOAP-ENC:base64Binary" />
  </all>
</complexType>
```

Sellisel juhul geneeritakse dto, mille küljest saab faili mugavalt kätte:

```
public interface IDocument {
    string Name {
        get;
        set;
    }
    System.IO.FileInfo Sisu {
        get;
        set;
    }
}
```