Enoncé complémentaire 2

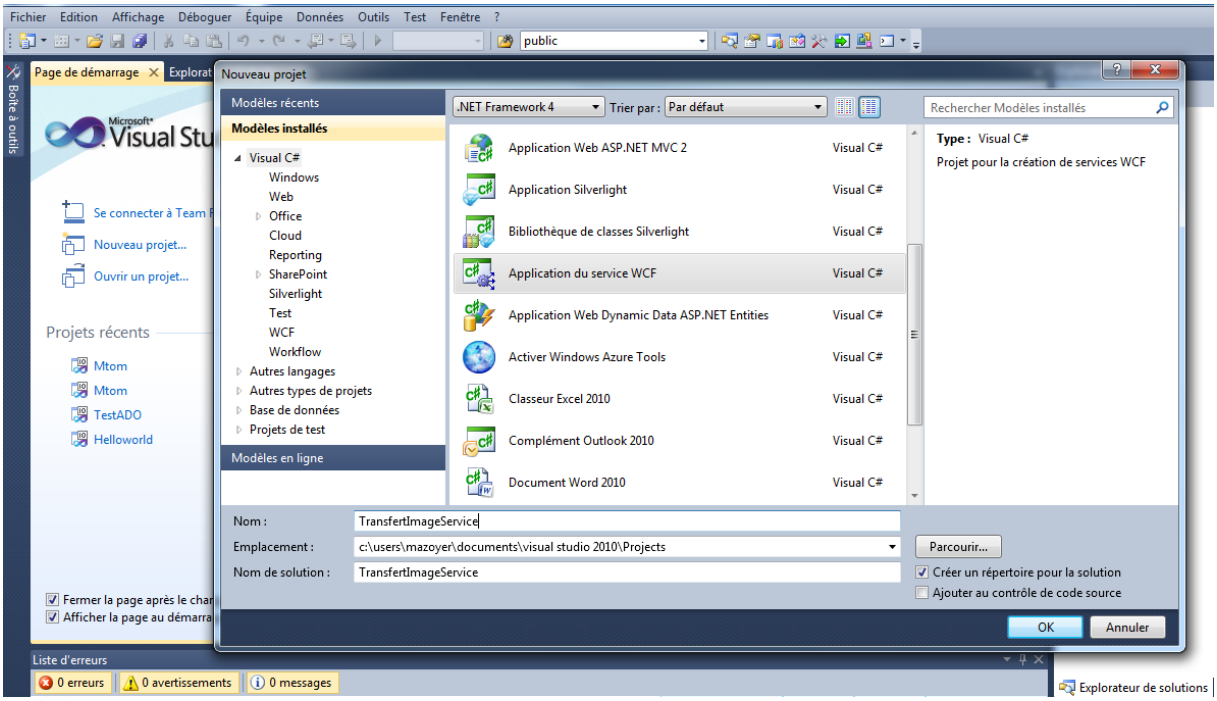
Tips :

1. Transférer des images avec un service WCF

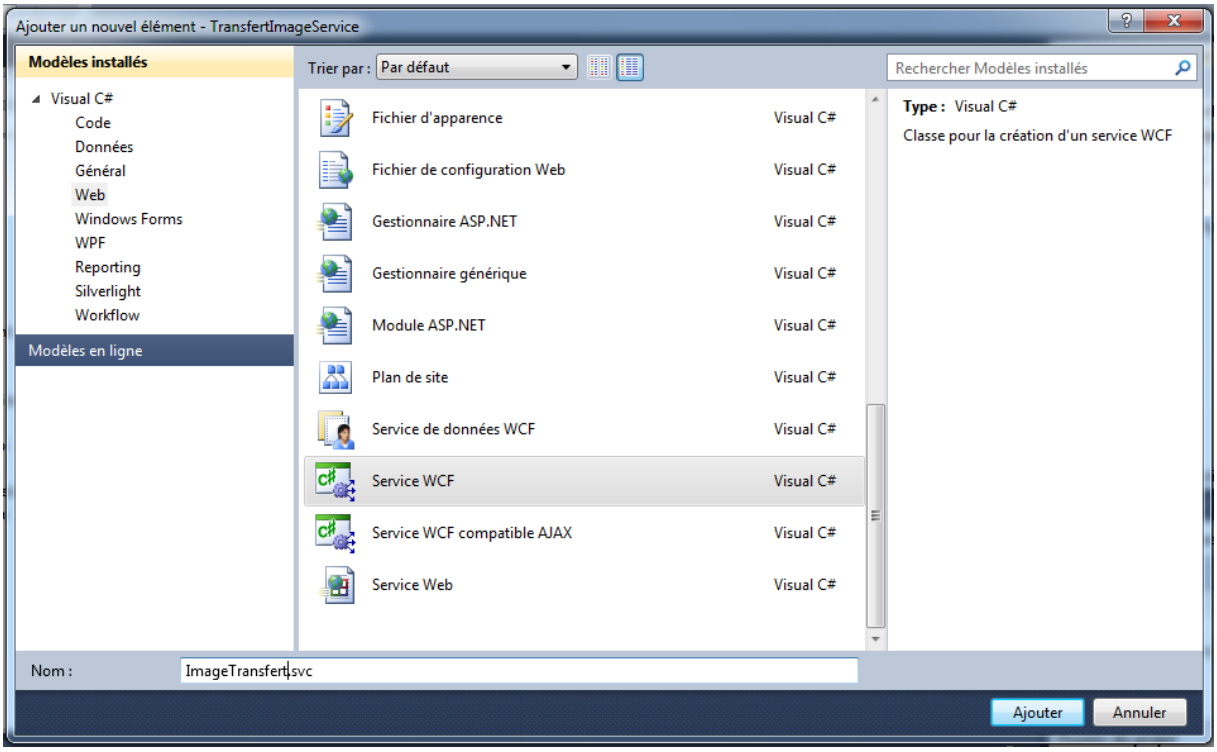
WCF (Windows Communication Foundation) est un composant du framework .NET qui permet de construire des applications distribuées sur le modèle SOA (Service Oriented Architecture). WCF permet de déployer des services et de les rendre accessibles à des clients à distance. WCF permet une communication sûre, fiable et implémente plusieurs protocoles, lui permettant ainsi d'interagir avec différentes briques logicielles. Dans cet exemple, nous allons voir comment implémenter un service WCF permettant à des clients de transférer des images sur le serveur.

1.1. Création du service et implémentation des méthodes

Créer un nouveau projet de type application de service WCF.



Ajouter un nouvel élément de type Service WCF :



Définir l'interface de notre service dans le fichier généré IImageTransfert.cs :

```
using System;
using System.ServiceModel;
using System.IO;

namespace TransfertImageService
{
    [ServiceContract]
    public interface IImageTransfert
    {
        [OperationContract]
        String UploadImage(Stream image);


        [OperationContract]
        Stream DownloadImage(String name);
    }
}
```

Implémenter dans le fichier FileTransfertService.svc les méthodes déclarées dans l'interface du service :

```
using System;
using System.IO;


namespace TransfertImageService
{
    public class ImageTransfert : IImageTransfert
    {


        // la classe AccesDonnees n'est pas donnée ici
        private AccesDonnees accesDonnees = new AccesDonnees();

        public String UploadImage(Stream image)
        {
            // Stocker l'image en BDD
            byte[] imageBytes = null;
            MemoryStream imageStreamEnMemoire = new MemoryStream();
            image.CopyTo(imageStreamEnMemoire);
            imageBytes = imageStreamEnMemoire.ToArray();
            String imageID = bdAccess.addImage(imageBytes);
            imageStreamEnMemoire.Close();
            image.Close();
            return imageID;
        }


        public Stream DownloadImage(String imageID)
        {
            // Récupérer l'image stockée en BDD et la transférer au client
            byte[] imageBytes = bdAccess.getImage(imageID);
            MemoryStream imageStreamEnMemoire = new MemoryStream(imageBytes);
            return imageStreamEnMemoire;
        }
    }
}
```

1.2. Configuration du service

Nous allons maintenant configurer le service en question. Nous allons changer deux paramètres à notre service WCF :

- Utilisation de la liaison WSHTTPBinding. Cette liaison, qui se base sur le protocole http pour le transport, est fiable et sécurisée. Elle supporte aussi les sessions.
- Utilisation de l'encodage MTOM pour les messages. Cet encodage est optimisé pour l'envoi des fichiers binaires par web Services en réduisant la taille des messages transmis.

Nous allons pour cela éditer le fichier Web.config du projet. On rajoute le code suivant dans la balise system.serviceModel :

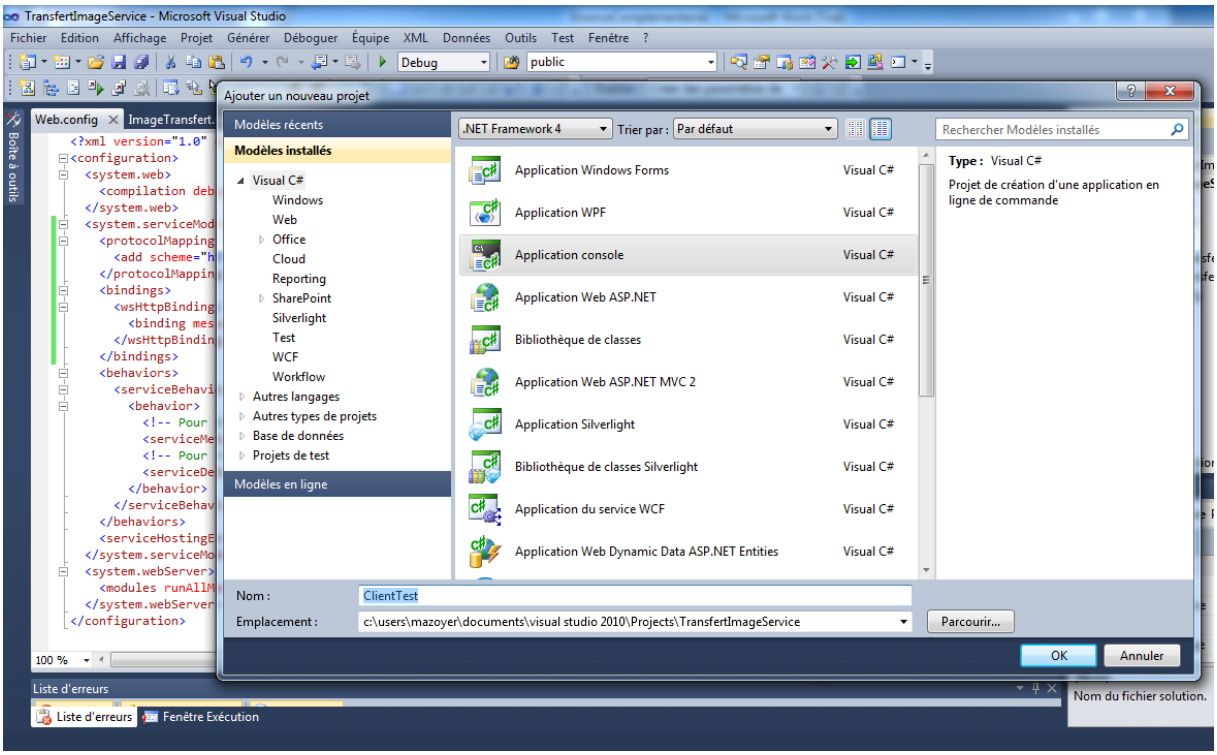
```
<protocolMapping>
  <add scheme="http" binding="wsHttpBinding" />
</protocolMapping>
<bindings>
  <wsHttpBinding>
    <binding messageEncoding="Mtom" maxReceivedMessageSize="10485760"/>
  </wsHttpBinding>
</bindings>
```

Nous allons chercher à récupérer le WDSL de notre service (clique-droit sur FileTransfertService.svc ->Afficher dans le navigateur) :

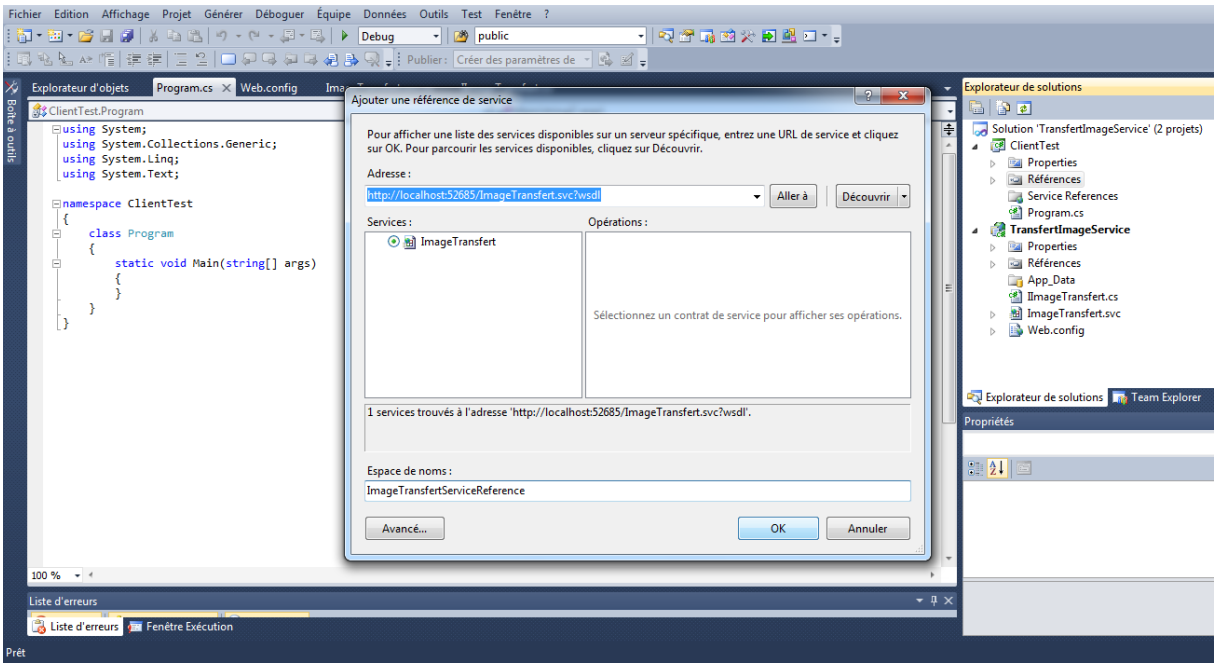
Le lien (http://adresse:port/ImageTransfert.svc?wsdl) donne accès à ce WSDL. Un fichier WSDL est un fichier XML qui contient la description des différentes méthodes du service ainsi que la location du service. Copier ce lien dans le presse-papier.

1.3. Création du client de test

On crée une application console pour tester notre service (clique-droit sur la solution-> Ajouter->Nouveau projet).



On ajoute une référence sur notre service WCF (clique-droit sur Référence->Ajouter une référence de service).



On spécifie l'adresse du WSDL de notre service, on clique sur « Aller à », on renomme la référence de service puis Ok.

Voici un exemple de code pour le client de test permettant d'invoquer la méthode UploadImage de notre service :

```
using System;
using System.IO;

namespace ClientTest
{
    class Program
    {
        static void Main(string[] args)
        {
            // Instanciation de la référence de service
            ImageTransfertServiceReference.ImageTransfertClient
            imageTransfertService = new
            ImageTransfertServiceReference.ImageTransfertClient();
            MemoryStream imageStream = new
            MemoryStream(lireFichier(@"c:\fichier.jpg"));
            // Appel de notre web method
            imageTransfertService.UploadImage(imageStream);
            Console.Out.WriteLine("Transfert Terminé");
            Console.ReadLine();
        }

        /// <summary>
        /// Lit et retourne le contenu du fichier sous la forme de tableau de byte
        /// </summary>
        /// <param name="chemin">chemin du fichier</param>
        /// <returns></returns>
        private static byte[] lireFichier(string chemin)
        {
            byte[] data = null;
            FileInfo fileInfo = new FileInfo(chemin);
            int nbBytes = (int)fileInfo.Length;
            FileStream fileStream = new FileStream(chemin, FileMode.Open,
            FileAccess.Read);
            BinaryReader br = new BinaryReader(fileStream);
            data = br.ReadBytes(nbBytes);
            return data;
        }
    }
}
```

1.4. Configuration du client

Dans le fichier app.config du client, on change la configuration du binding afin d'augmenter la taille maximum des messages, des buffers ainsi que les différents timeout afin que la connexion supporte le transfert de fichiers :

```
<binding name="WSHttpBinding_IImageTransfert" closeTimeout="00:10:00"
                openTimeout="00:10:00" receiveTimeout="01:00:00"
sendTimeout="01:00:00"
                bypassProxyOnLocal="false" transactionFlow="false"
hostNameComparisonMode="StrongWildcard"
                maxBufferPoolSize="1085760" maxReceivedMessageSize="1310720"
                messageEncoding="Mtom" textEncoding="utf-8"
useDefaultWebProxy="true"
                allowCookies="false">
              <readerQuotas maxDepth="1024" maxStringContentLength="163840"
maxArrayLength="327680"
                  maxBytesPerRead="131072" maxNameTableCharCount="327680" />
              <reliableSession ordered="true" inactivityTimeout="00:10:00"
                  enabled="false" />
              <security mode="Message">
                <transport clientCredentialType="Windows"
proxyCredentialType="None"
                    realm="" />
                <message clientCredentialType="Windows"
negotiateServiceCredential="true"
                    algorithmSuite="Default" />
              </security>
</binding>
```

Le service est maintenant prêt à être testé. Définir le projet de test comme projet de démarrage (clique-droit sur le projet->Définir comme projet de démarrage) et démarrer le projet.

Attention : l'exécution de la première web méthode peut paraître assez longue : en effet, c'est lors du premier appel de méthode que la connexion est établie. Les appels de web méthodes suivants se feront beaucoup plus vite.

1.5. Pour aller plus loin

1.5.1. Streaming

Tel que l'on a implémenté le service, on a utilisé le type Stream pour transférer le contenu de notre image. Ce type permet aussi de transférer les fichiers en streaming. Le streaming peut être utile dans le cas de transfert de gros fichier afin d'économiser de la mémoire vive sur le serveur. Pour cela il faudrait rajouter l'attribut transferMode à « Streamed » dans la configuration du binding. Néanmoins, il faudrait pour cela utiliser un basicHttpBinding puisque le wsHttpBinding (liaison fiable et sécurisé) ne supporte pas le streaming.

1.5.2. Utilisation des DataContract

La définition de DataContract permet de transférer des objets personnels avec notre service WCF. Les objets d'un DataContract doivent être serializable. Dans notre exemple, on transfert un Stream dans notre message. Il est impossible de transférer d'autres paramètres (comme l'ID de notre image) dans le corps du message lorsqu'on utilise un Stream. On va donc voir comment créer un objet DataContract contenant les infos sur notre image et le transférer dans l'en-tête de notre message. Voici le code de notre interface, des objets MessageContract contenant la définition des messages échangés ainsi que l'objet DataContract qui contient les informations sur notre image que l'on souhaite transférer en en-têtes avec notre fichier. (Il faut bien sûr modifier le code du client pour l'appel du service ainsi que l'implémentation du service lui-même en fonction de la nouvelle interface) :

```
using System;
using System.IO;
using System.Collections.Generic;
using System.Runtime.Serialization;
using System.ServiceModel;

namespace FileTransfertService
{


    [ServiceContract]
    public interface IFileTransfertService
    {
        [OperationContract]
        void Upload(ImageUploadRequest data);


        [OperationContract]
        ImageDownloadResponse Download(ImageDownloadRequest data);
    }

    [MessageContract]
    public class ImageUploadRequest
    {
        [MessageHeader(MustUnderstand = true)]
        public ImageInfo ImageInfo;

        [MessageBodyMember(Order = 1)]
        public Stream ImageData;


    }

    [MessageContract]
    public class ImageDownloadResponse
    {

        [MessageBodyMember(Order = 1)]
        public Stream ImageData;
    }

    [MessageContract]
    public class ImageDownloadRequest
    {

        [MessageBodyMember(Order = 1)]
        public ImageInfo ImageInfo;
    }


    [DataContract]
    public class ImageInfo
    {
        [DataMember(Order = 1, IsRequired = true)]
        public string ID { get; set; }

        [DataMember(Order = 2, IsRequired = true)]
        public string Album { get; set; }
    }
}
```