

# Développement en c#.NET

**Ing. Meryem OUARRACHI**

# Plan du module

## Langage C#

- ☐ L'environnement .Net
- ☐ Initiation à la programmation C#
- ☐ Programmation Orienté Objet C#

## Programmation avancée en .Net ,C#

- ☐ **Programmation distribuée**
- ☐ Gestion de base de donnée
- ☐ Application WPF

## Chapitre 6:

# WEB SERVICE

# Les Web services

- L'informatique réparti implique:

- Une application distribuée c à d d'autres applications peuvent y accéder;

- Une architecture adaptative et ouverte→ cela nécessite le respect des normes **SOA**-Service Oriented Architectures-(séparation de l'interface client et la gestion métier)

→Même application de traitement et plusieurs applications de la mise en forme

# Les Web services

- Le but de Web service est de créer une application et la rendre disponible à plusieurs type de clients.

→ Mettre en place une architecture **indépendante** de plateforme utilisée. *Exemple:*

- Etre un client en Windows et utiliser une application créée en linux.

- Lors de travail en java, on peut appeler une application écrite en C#, php...

- Utiliser la même application dans un environnement desktop, mobile, web

# Les Web services

- Les services web permettent d'adapter les architectures distribuées dans le web.
- **Le services WCF** est le nom de service web en .Net, correspond à des classes contenant des méthodes exposés en réseau afin que d'autres applications puissent les utiliser.

# Services web :Architecture



- Les échanges sont codés sous la forme de message **SOAP**: *Simple Object Acces Protocol*.
- C'est un protocole qui assure les appels des procédures à distance au dessus d'un protocole de transport(http)

# Message SOAP

- Les messages SOAP doivent respecter une syntaxe XML.
- Les paramètres d'entrée sont sérialisées dans le corps en respectant une syntaxe XML.
- Le retour est de même sérialisé en XML.
- Paramètres d'entrée et retour peuvent être des objets composés.



# Message SOAP

Dans le web service on a Requête SOAP et Réponse SOAP :

- Le client invoque une demande: cette demande se transforme de langage utilisé pour le codage en code XML(Sérialisation) on parle de requête SOAP;
- Le serveur reçoit cette requête et déséréalise le message afin d'exécuter la méthode appelée par le client. Après le résultat est sérialisé et envoyé au client (Réponse SOAP)

*Remarque:* En .Net, la classe « XML Serializer » responsable de la sérialisation et déséréalisation

# Les étapes de mise en œuvre de service web

## **Etape1:** Création de service web

- Définition du contrat
- Implémentation du contrat
- Configuration du service

## **Etape2:** consommation de service web

- Création d'une application cliente

# Création de service web

**1.définition du contrat:** On définit les signatures des méthodes composant notre service WCF, et du format des données à échanger.

- Pour le définir on utilise les métadonnées suivantes :

- ServiceContract:** indique que la classe ou l'interface est un contrat de service.

- OperationContract:** attachée aux méthodes que l'on souhaite exposer au travers du service WCF.

- DataContract/DataMembre:** décrit les types de données utilisées par un service.

# Création de service web

## 1.définition du contrat:

Créer un nouveau projet de type « Service WCF Application »

```
namespace WcfService2
{
    [ServiceContract]
    public interface IService1
    {
        [OperationContract]
        string GetData(int value);

        [OperationContract]
        CompositeType GetDataUsingDataContract(CompositeType composite);
    }

    [DataContract]
    public class CompositeType
    {
        bool boolValue = true;
        string stringValue = "Hello ";

        [DataMember]
        public bool BoolValue
        {
            get { return boolValue; }
            set { boolValue = value; }
        }

        [DataMember]
        public string StringValue
        {
            get { return stringValue; }
            set { stringValue = value; }
        }
    }
}
```

# Création de service web

**2.Implémentation du contrat :** On implémente les services.

**3.Configuration du service:** On définit les endPoints, autrement dit les points d'accès au service.

- Les éléments A B C**

- **Adresse** : adresse à laquelle le client doit se connecter pour utiliser le service.
- **Binding** : protocole à utiliser par le client pour communiquer avec le service.
- **Contrat** : infos échangées entre le serveur et le client afin que ce dernier sache comment utiliser le service.

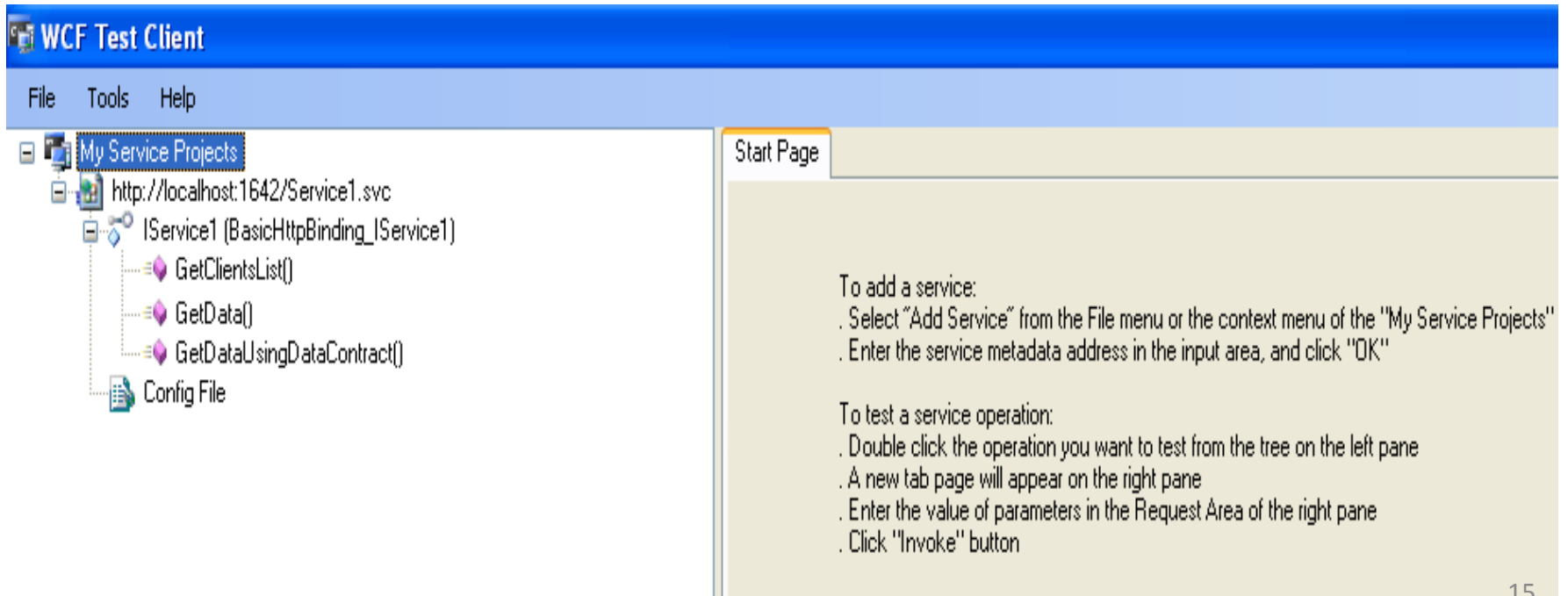
# Web Service Description Language

## WSDL

- Après ces étapes, on a un fichier qui se génère automatiquement appelé WSDL, il s'agit d'un document XML qui décrit le détail de service.
- Ce fichier décrit les méthodes et les paramètres attendu. S'il s'agit d'objets complexes il décrit leurs structures.
- En lisant le fichier WSDL le client sait ce que le serveur peut faire.

# Client test WCF

-C'est un outil graphique permet aux utilisateurs d'entrer des paramètres de test, d'envoyer ses entrées au service et d'afficher la réponse renvoyée par ce dernier.



# Client test WCF

-C'est un outil graphique permet aux utilisateurs d'entrer des paramètres de test, d'envoyer ses entrées au service et d'afficher la réponse renvoyée par ce dernier.

The screenshot displays the WCF Client Test tool interface. It features a tab labeled 'GetData'. Below the tab, there are two main sections: 'Request' and 'Response'. The 'Request' section contains a table with three columns: 'Name', 'Value', and 'Type'. The first row shows 'value' in the 'Name' column, '5' in the 'Value' column, and 'System.Int32' in the 'Type' column. The 'Response' section also contains a table with three columns: 'Name', 'Value', and 'Type'. The first row shows '(return)' in the 'Name' column, '"You entered: 5"' in the 'Value' column, and 'System.String' in the 'Type' column. To the right of the 'Response' table, there is a checkbox labeled 'Start a new proxy' and a button labeled 'Invoke'.

Name	Value	Type
value	5	System.Int32

Name	Value	Type
(return)	"You entered: 5"	System.String

☐ Start a new proxy



# Consommation de service web

- Afin de consommer un service il faut générer une **classe proxy** .

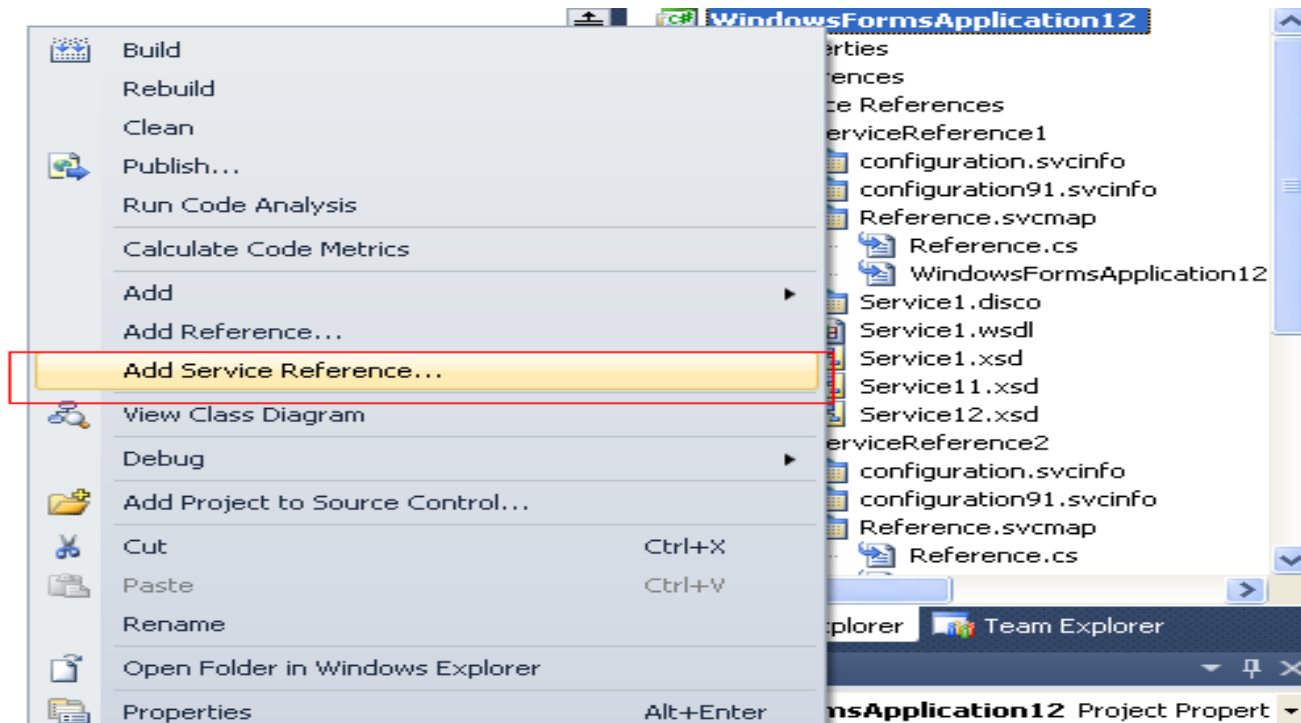
- Le rôle de cette classe est d'exposer les méthodes proposées par le service distant. Il s'agit d'un intermédiaire entre le client et le service.

# Consommation de service web

Afin de générer la classe proxy :

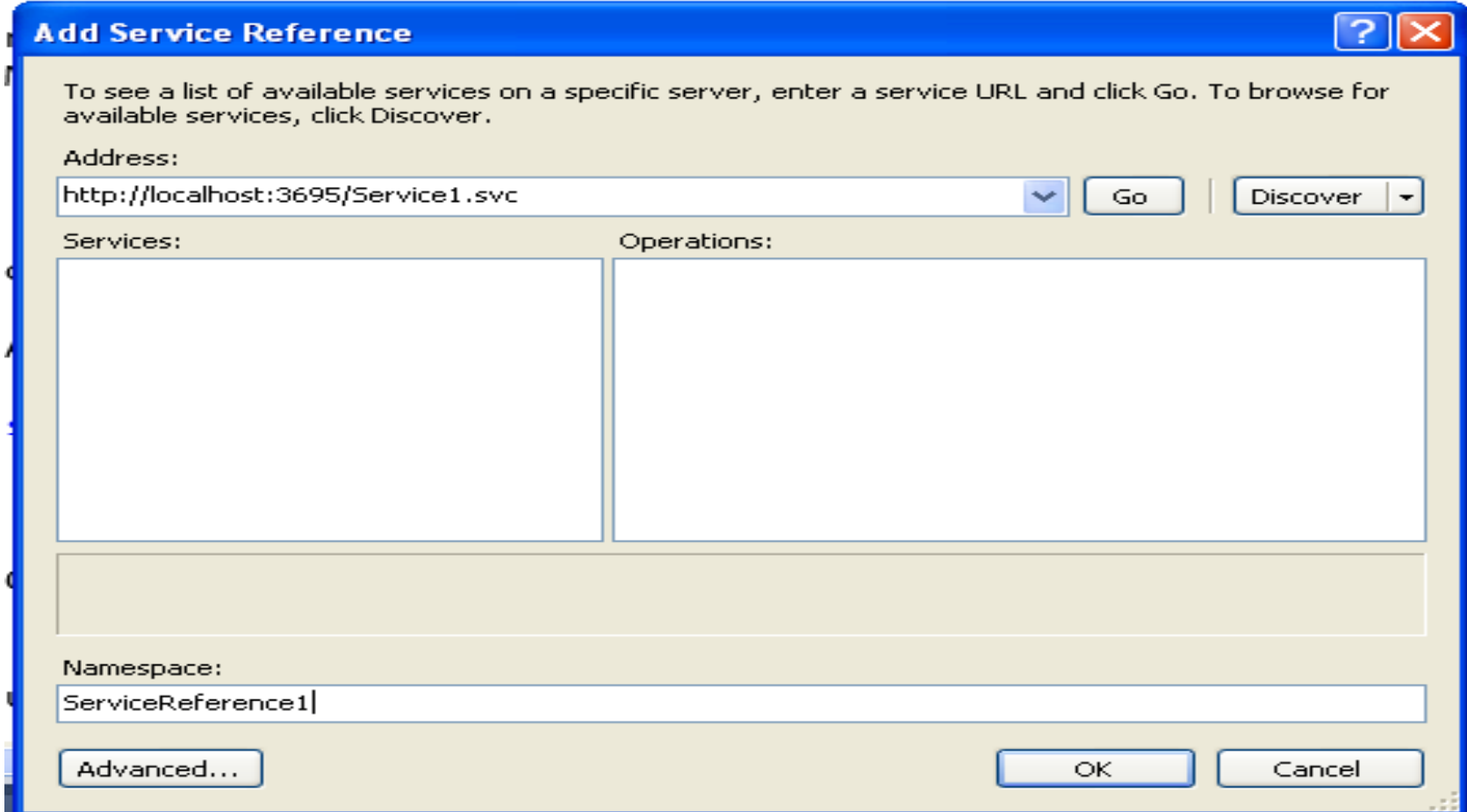
Dans le projet client, vous devez ajouter une référence vers le service web, utiliser l'assistant.

-Donner un nom significatif à votre référence web.



# Consommation de service web

Afin de générer la classe proxy :



The image shows a Windows-style dialog box titled "Add Service Reference". It has a blue title bar with a question mark icon and a close button. The main area has a light beige background. At the top, there is instructional text: "To see a list of available services on a specific server, enter a service URL and click Go. To browse for available services, click Discover." Below this, there is an "Address:" label followed by a text box containing "http://localhost:3695/Service1.svc". To the right of the text box are two buttons: "Go" and "Discover". Below the "Address:" section, there are two empty rectangular boxes labeled "Services:" and "Operations:". At the bottom, there is a "Namespace:" label followed by a text box containing "ServiceReference1". At the very bottom, there are three buttons: "Advanced...", "OK", and "Cancel".

**Add Service Reference**

To see a list of available services on a specific server, enter a service URL and click Go. To browse for available services, click Discover.

Address:

http://localhost:3695/Service1.svc

Go Discover

Services: Operations:

Namespace:

ServiceReference1

Advanced... OK Cancel

# Consommation de service web

-Dans votre code client:

//récupérer le proxy

**NomReference.NomProxy leProxy;**

**leProxy = new NomReference.NomProxy ();**

//appeler le service web

**TextBox1.Text = leProxy.HelloWorld();**

# Hosting de service web

-On a trois manière pour héberger le service web:

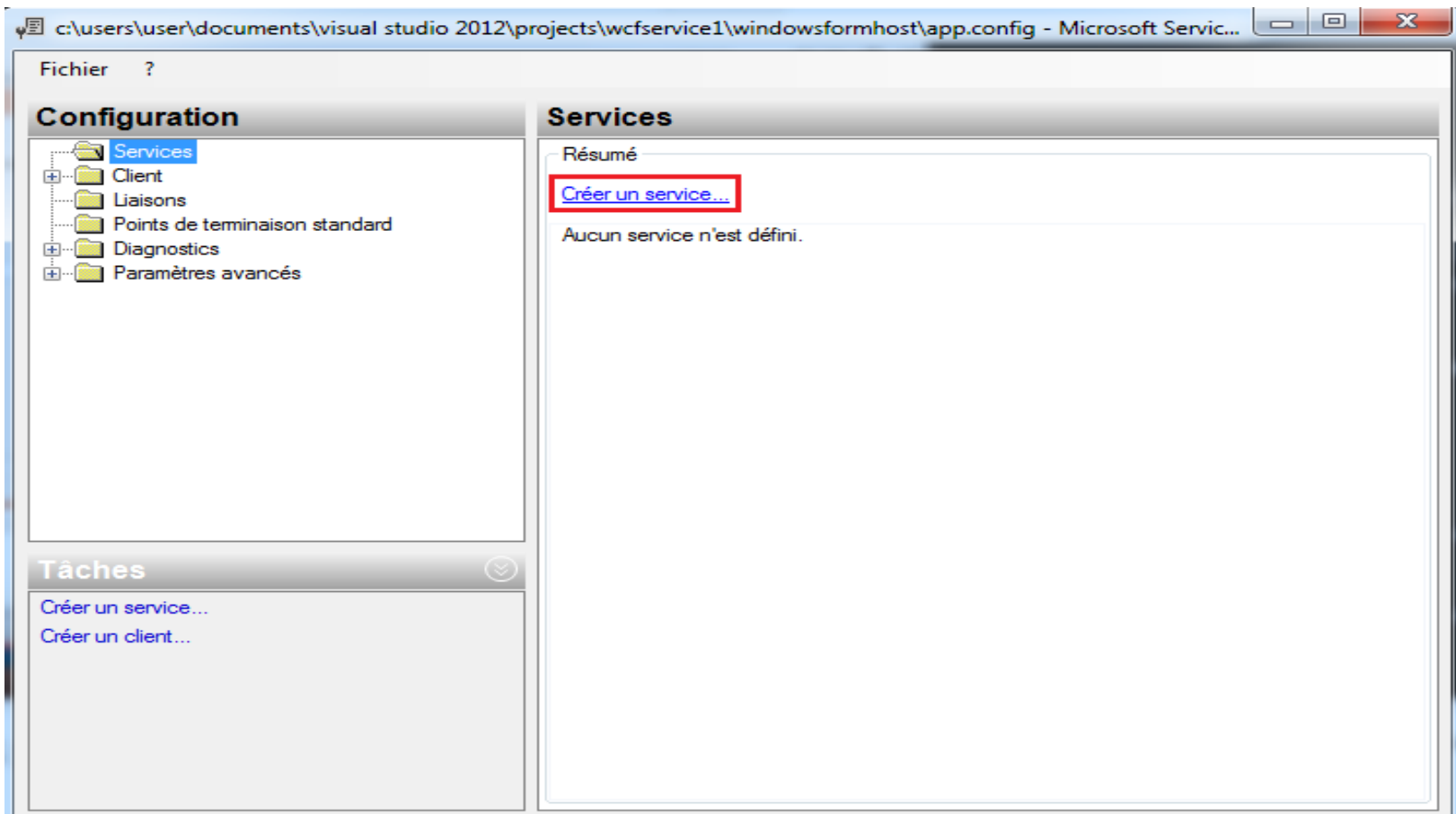
**1.Self hosting:** manuellement via un projet

**2.Windows Service:** intégré dans les services windows

**3.Hosting Server(IIS)**

# Self Hosting

Click droit sur App.config → modifier la configuration WCF



# Self Hosting

Déterminer le service à héberger

Assistant Nouvel élément de service

**Quel contrat de service utilisez-vous ?**

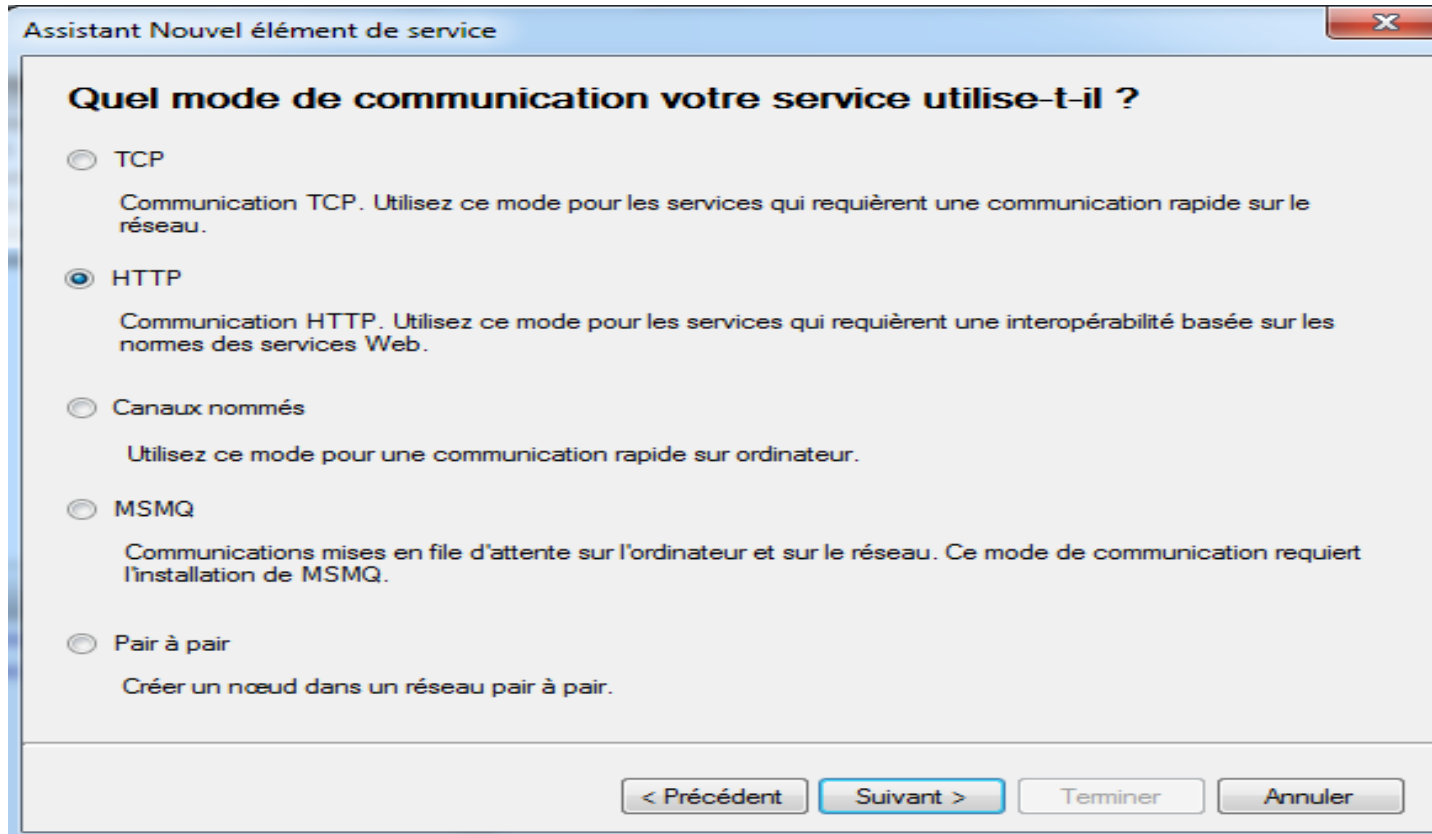
Un contrat est un regroupement d'opérations qui spécifie ce que le point de terminaison communique au monde extérieur. Il s'agit généralement du nom de l'interface défini dans le code. Vous pouvez utiliser l'option "Parcourir..." pour sélectionner le contrat à partir de votre assembly de service.

Spécifiez le contrat de service pour le service :

Contrat :

# Self Hosting

## Le protocole de communication



The screenshot shows a Windows XP-style dialog box titled "Assistant Nouvel élément de service". The main question is "Quel mode de communication votre service utilise-t-il ?". There are five radio button options, each with a descriptive text block below it. The "HTTP" option is selected. At the bottom, there are four buttons: "< Précédent", "Suivant >", "Terminer", and "Annuler".

**Assistant Nouvel élément de service**

**Quel mode de communication votre service utilise-t-il ?**

- ☐ TCP  
Communication TCP. Utilisez ce mode pour les services qui requièrent une communication rapide sur le réseau.
- ☒ HTTP  
Communication HTTP. Utilisez ce mode pour les services qui requièrent une interopérabilité basée sur les normes des services Web.
- ☐ Canaux nommés  
Utilisez ce mode pour une communication rapide sur ordinateur.
- ☐ MSMQ  
Communications mises en file d'attente sur l'ordinateur et sur le réseau. Ce mode de communication requiert l'installation de MSMQ.
- ☐ Pair à pair  
Créer un nœud dans un réseau pair à pair.

< Précédent   Suivant >   Terminer   Annuler



# Self Hosting

## -La sécurité à utiliser

Assistant Nouvel élément de service

**Quelle méthode d'interopérabilité voulez-vous utiliser ?**

☒ **Interopérabilité de base des services Web**  
Cette option permet l'interopérabilité avec les services Web qui implémentent les spécifications WS-I Basic Profile et Basic Security Profile. Choisissez cette option si vous communiquez avec les services Web ASP.NET ou d'autres services Web conformes à WS-I Basic Profile et Basic Security Profile.

☐ **Interopérabilité avancée des services Web**  
Cette option permet l'interopérabilité avec les services Web qui prennent en charge les transactions distribuées et les sessions fiables et sécurisées. Choisissez cette option si vous communiquez avec Windows Communication Foundation (WCF) ou d'autres services Web conformes aux spécifications WS-\*.

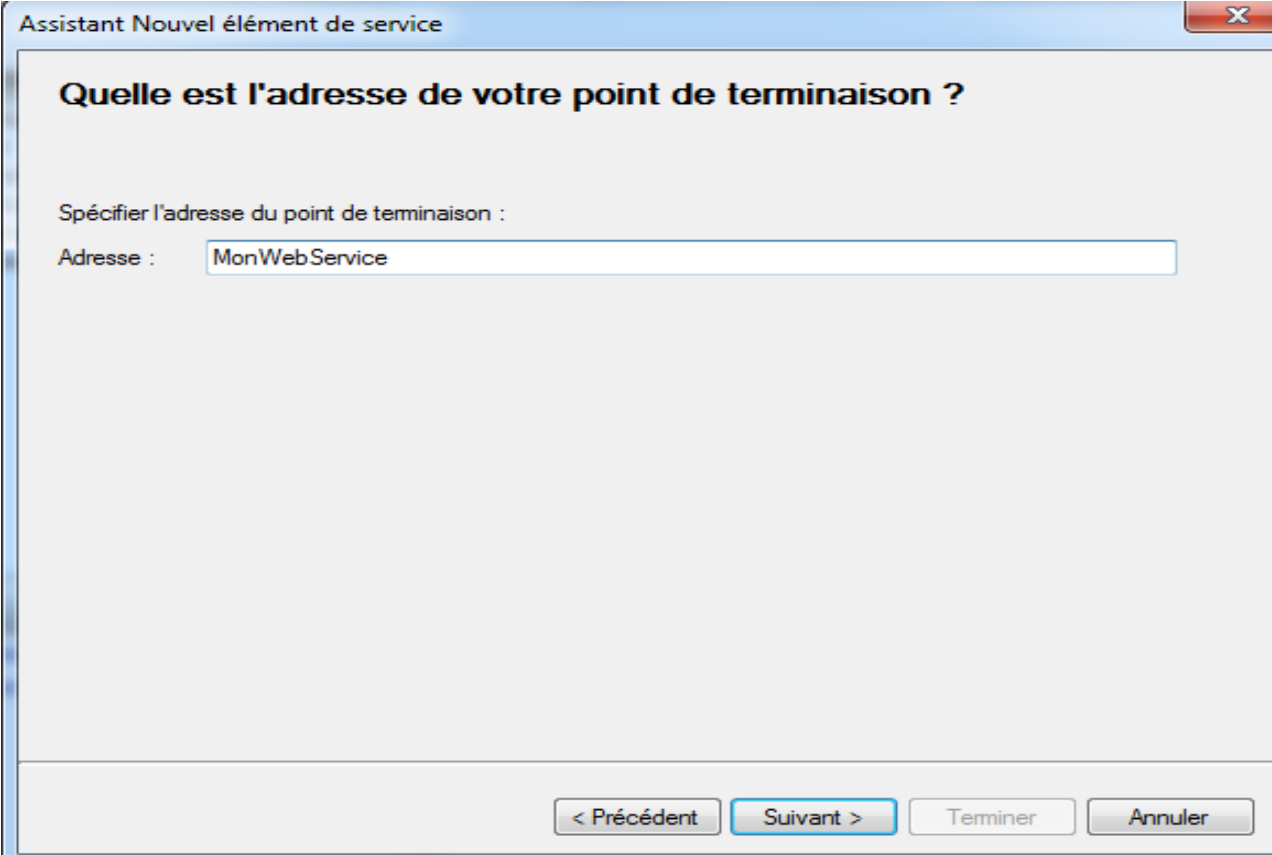
☐ **Communication simplex**  
Le contrat de service utilise la communication unidirectionnelle.

☐ **Communication duplex**  
Cette option active le contrat de service pour l'envoi et la réception de messages.

< Précédent   Suivant >   Terminer   Annuler

# Self Hosting

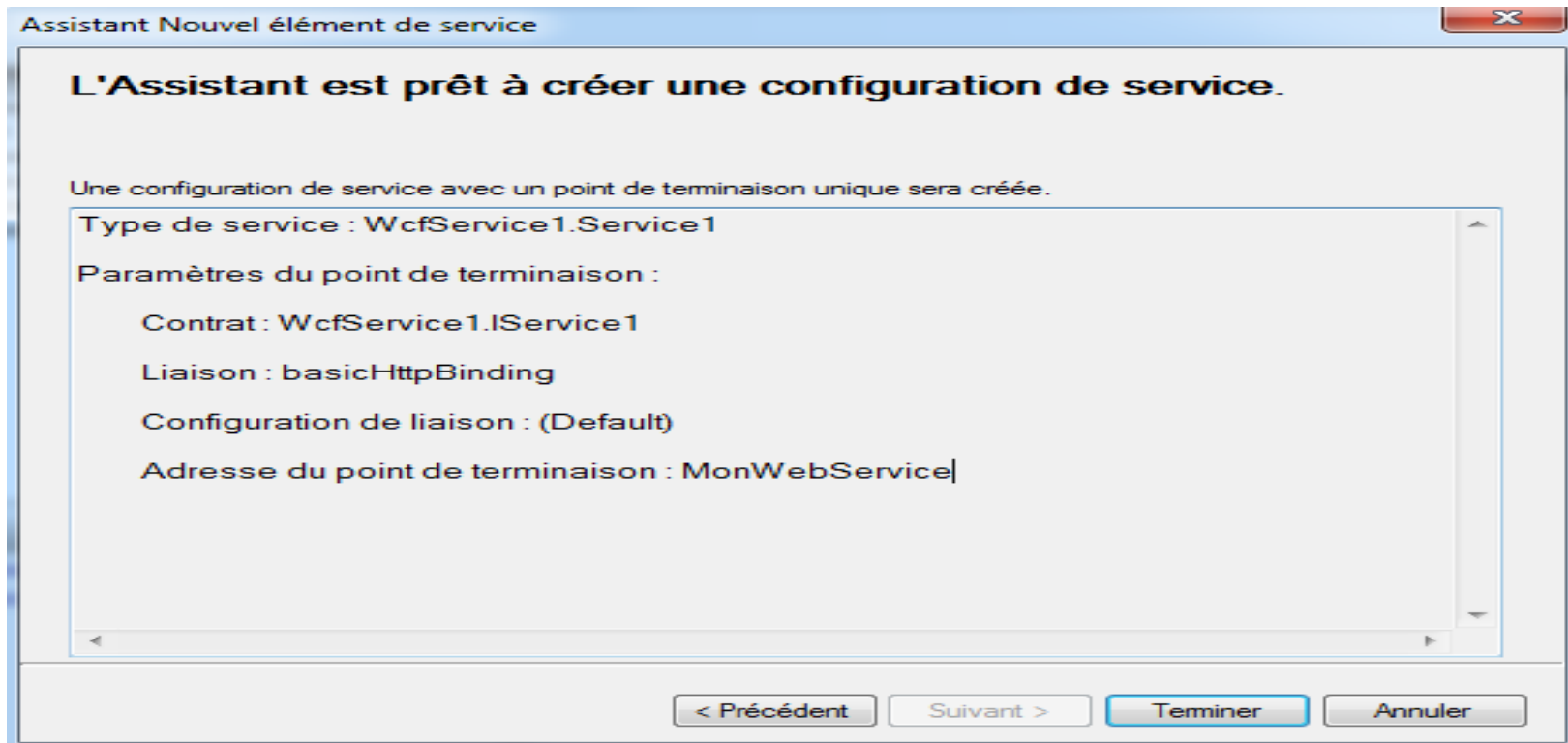
-Mettez n'importe quel nom, par la suite on détermine l'adresse correcte



The image shows a Windows-style dialog box titled "Assistant Nouvel élément de service". The main text inside the dialog asks "Quelle est l'adresse de votre point de terminaison ?". Below this, there is a label "Spécifier l'adresse du point de terminaison :" followed by a text input field. The input field contains the text "MonWebService". At the bottom of the dialog, there are four buttons: "< Précédent", "Suivant >" (which is highlighted with a blue border), "Terminer", and "Annuler".

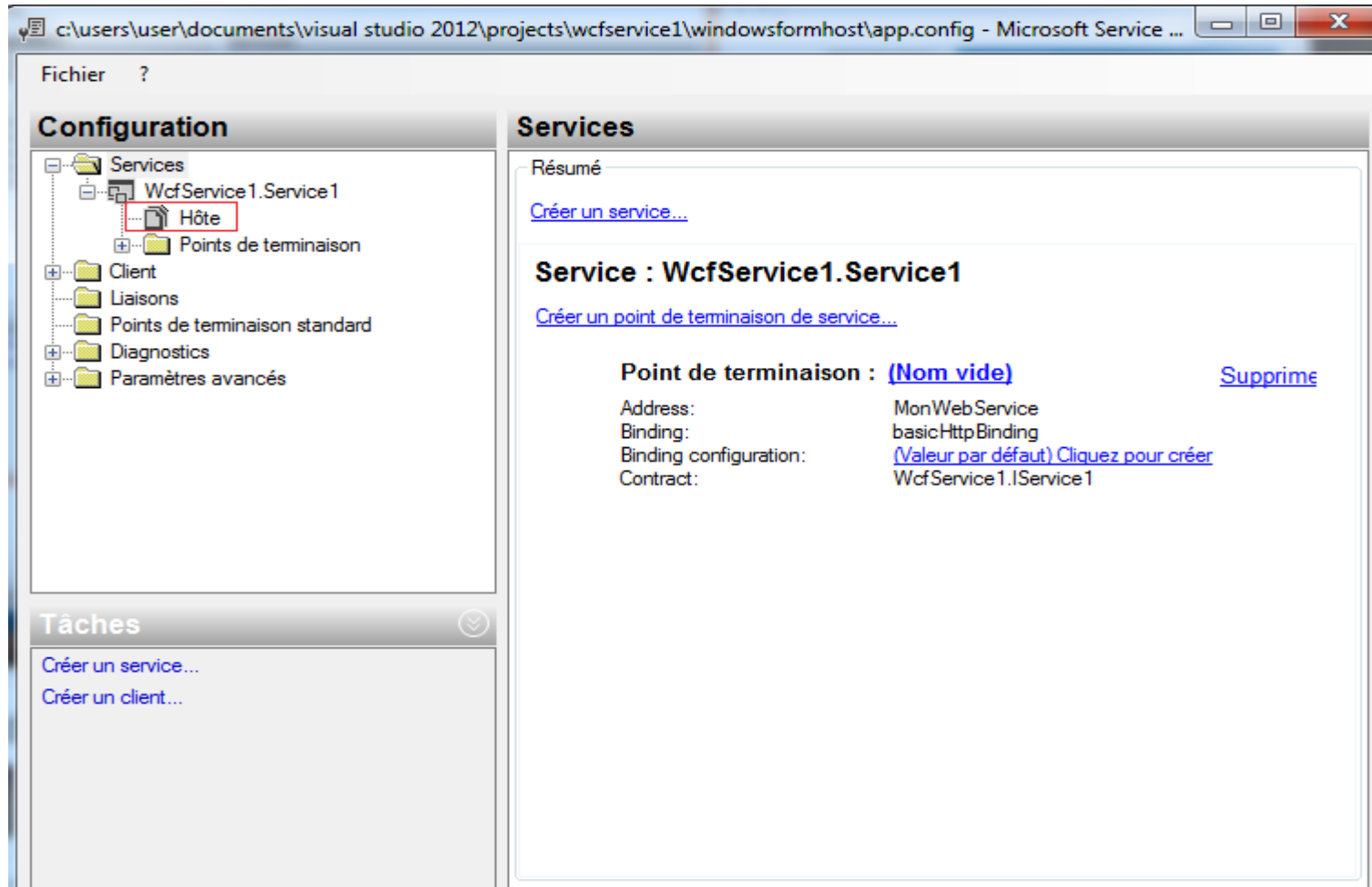
# Self Hosting

-La configuration est établie



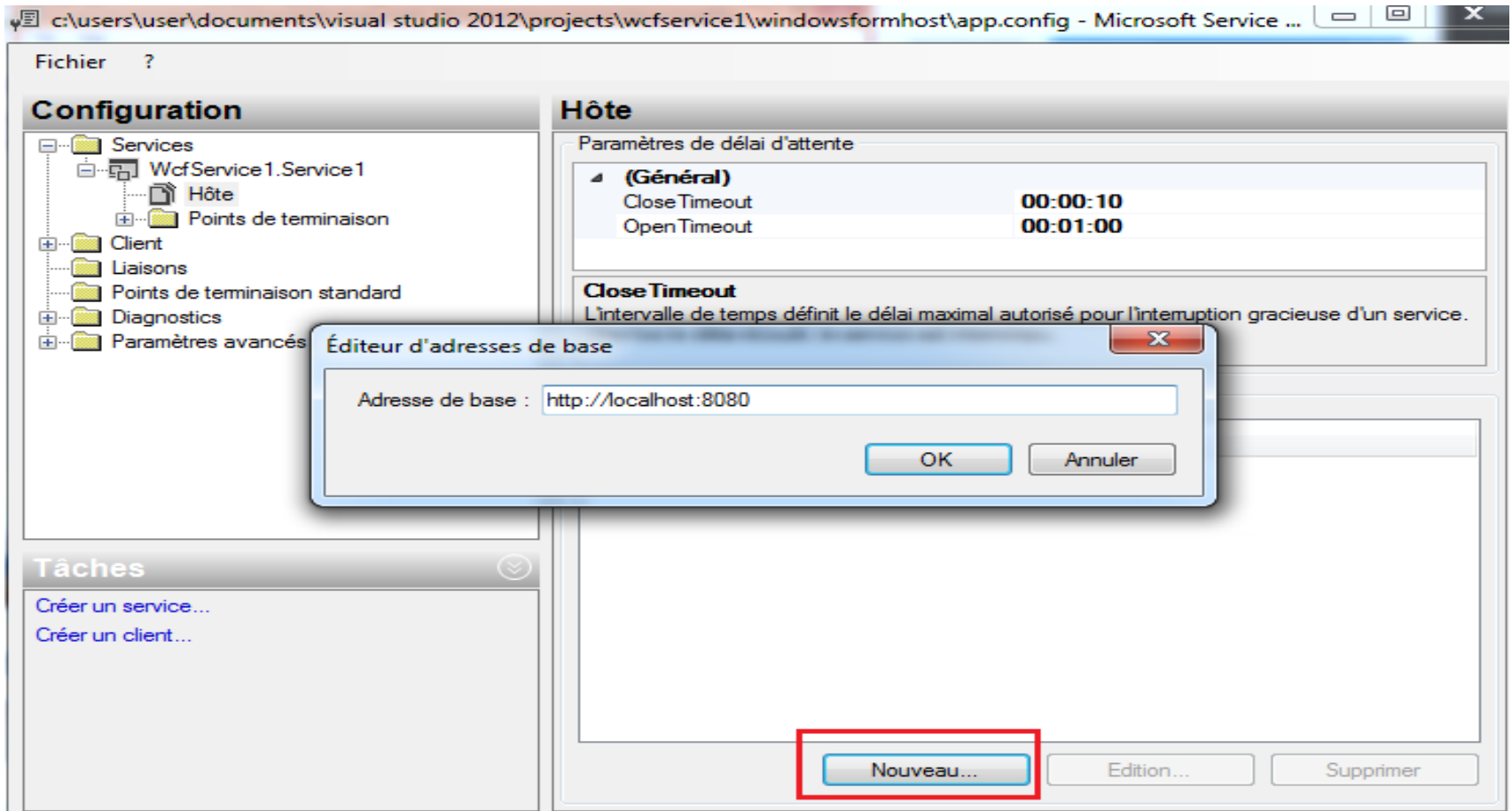
# Self Hosting

-La configuration est établie



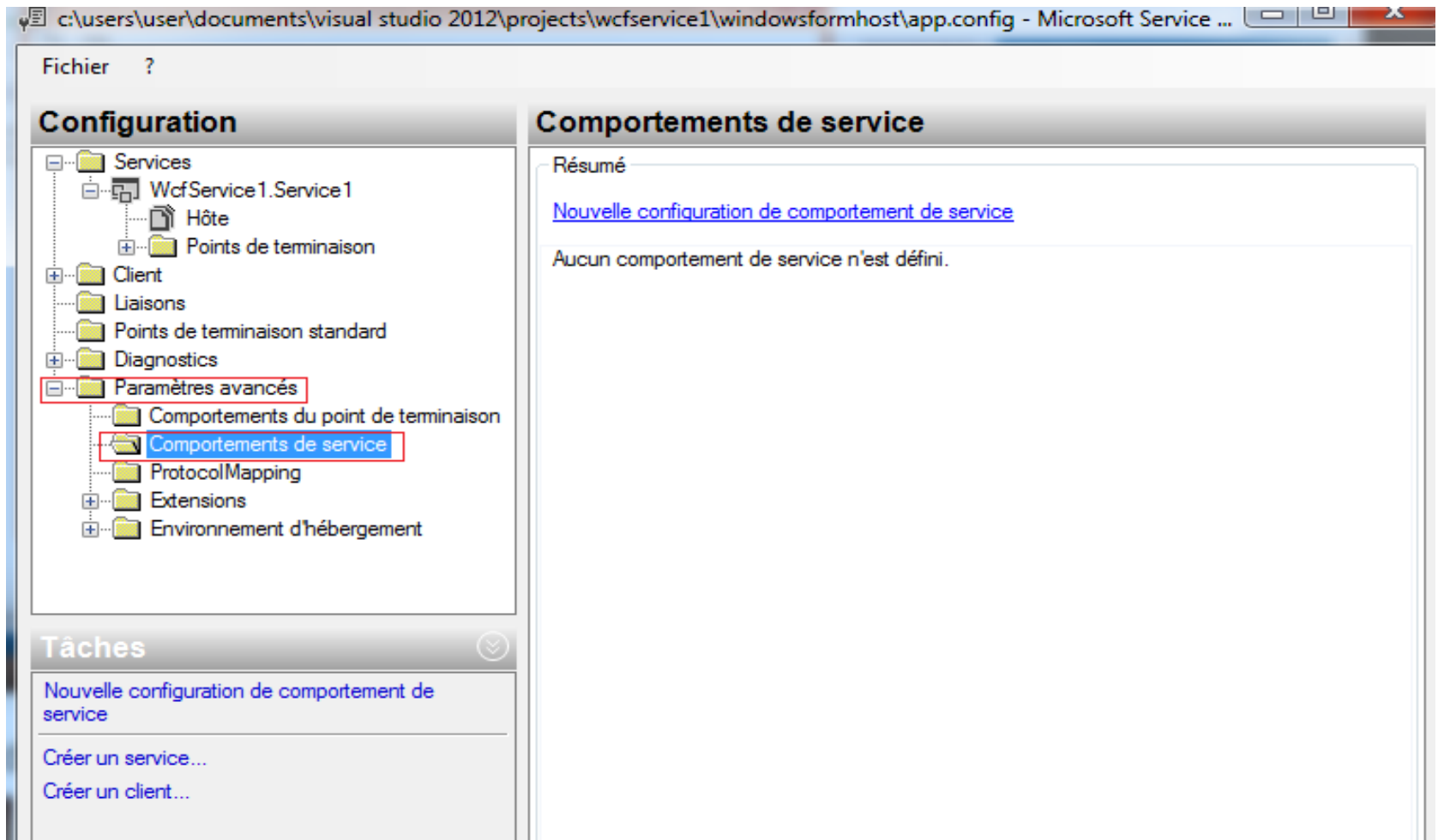
# Self Hosting

-Nouveau hôte: adresse qui va être utiliser par le client



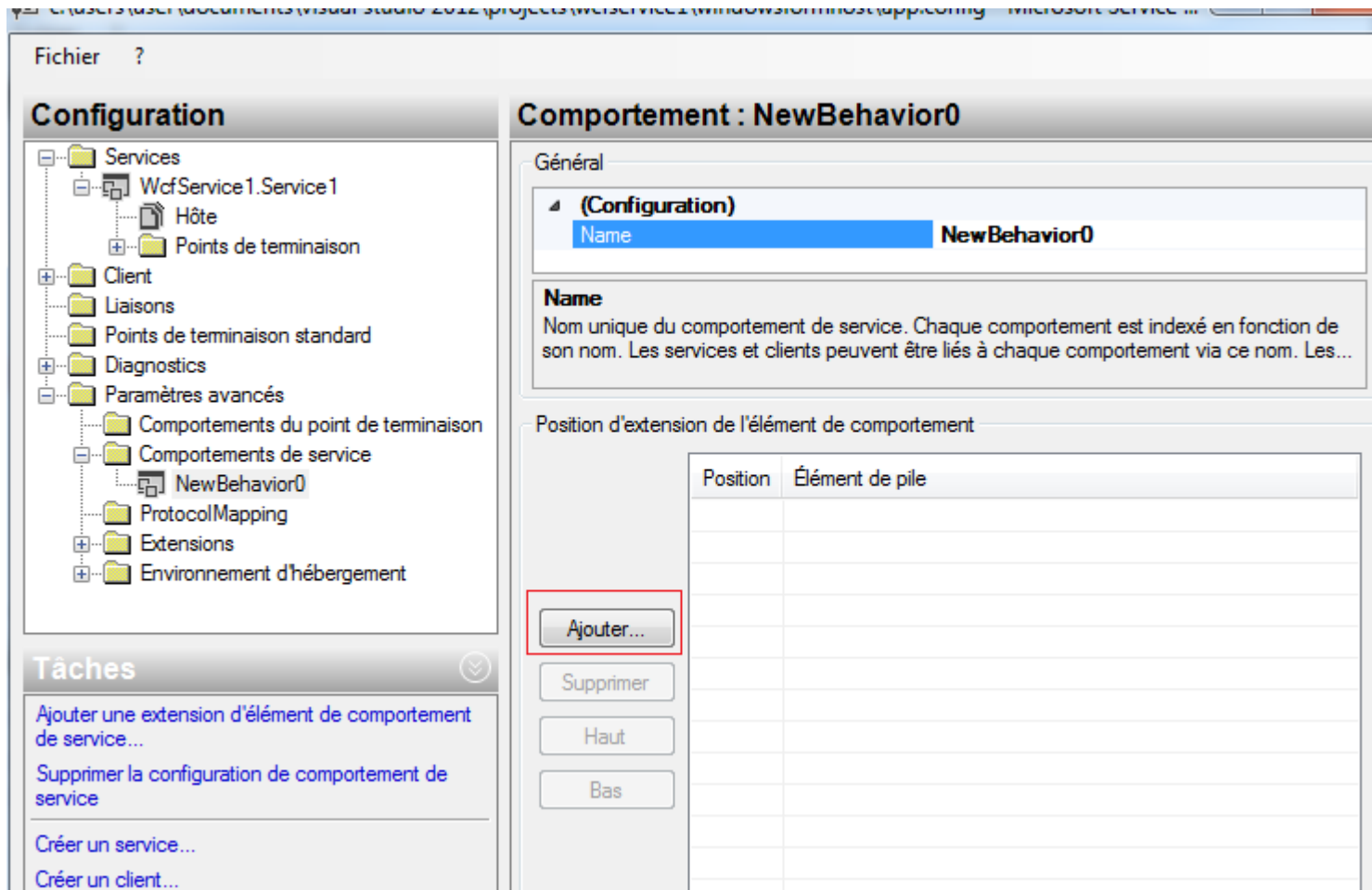
# Self Hosting

## -Configuration de description de service



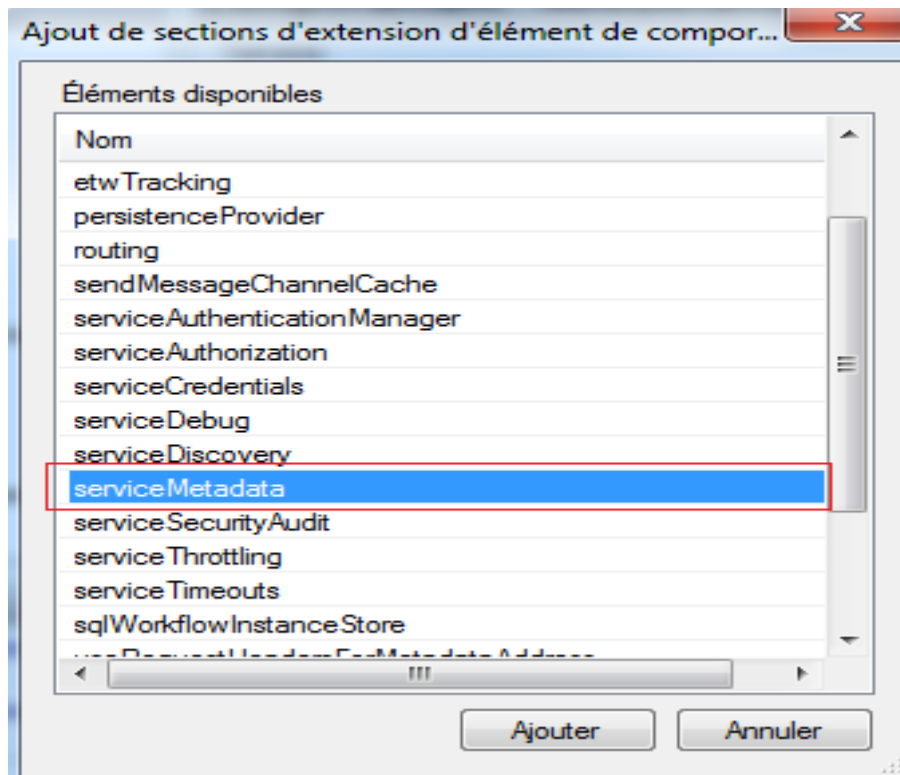
# Self Hosting

## -Configuration de description de service



# Self Hosting

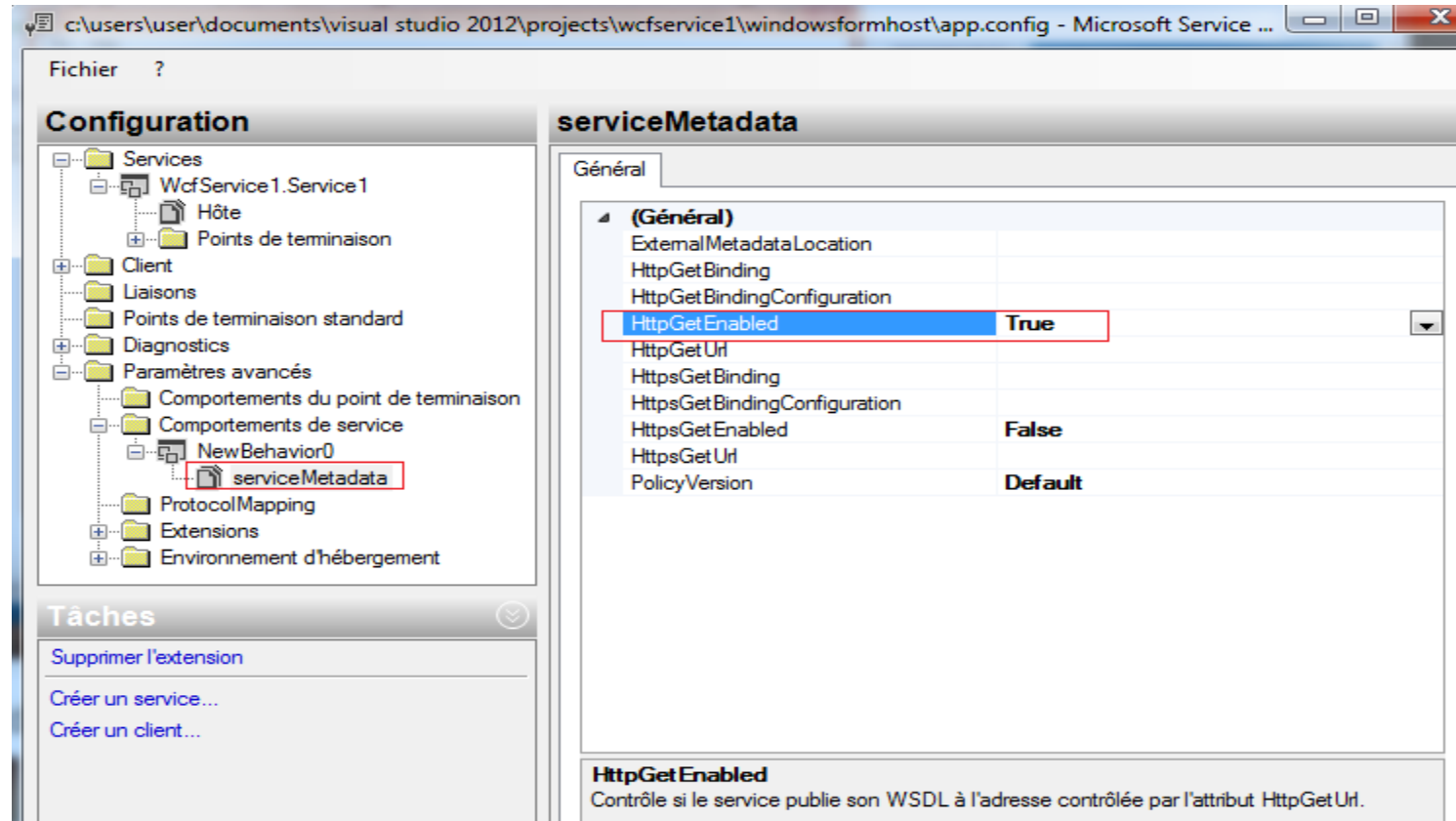
## Configuration de description de service





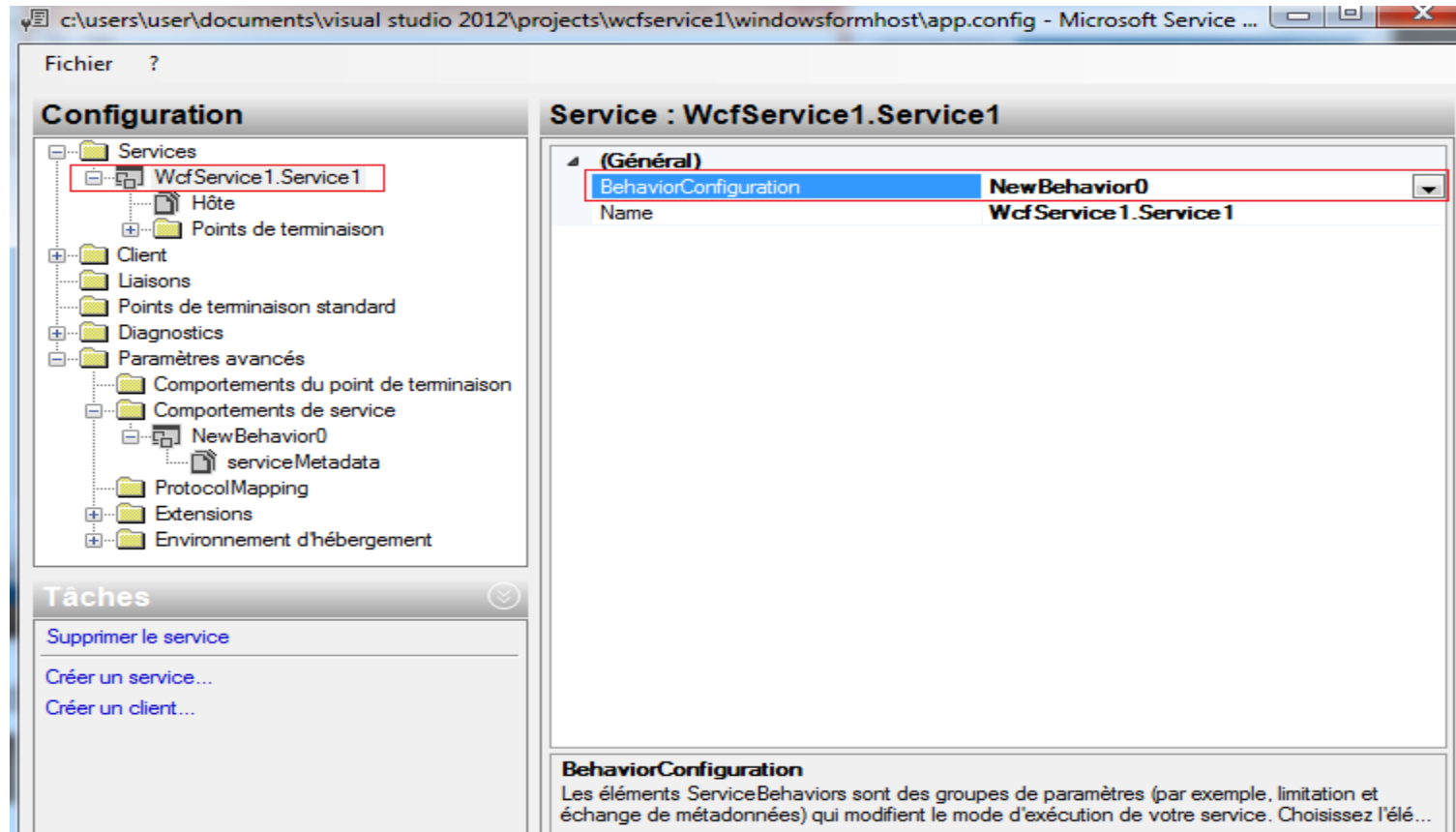
# Self Hosting

## -Configuration de description de service



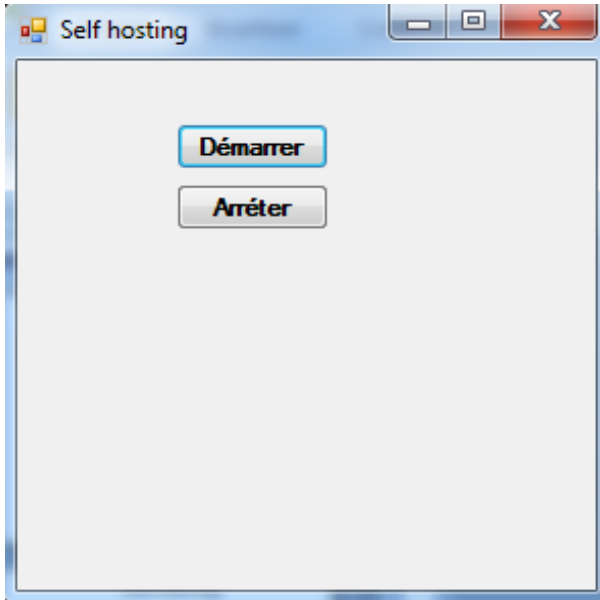
# Self Hosting

-Lier le service web à cette description



# Self Hosting

**-ServiceHost:** permet d'héberger le service, il va charger la classe de service.



**-Pour démarrer le service:**

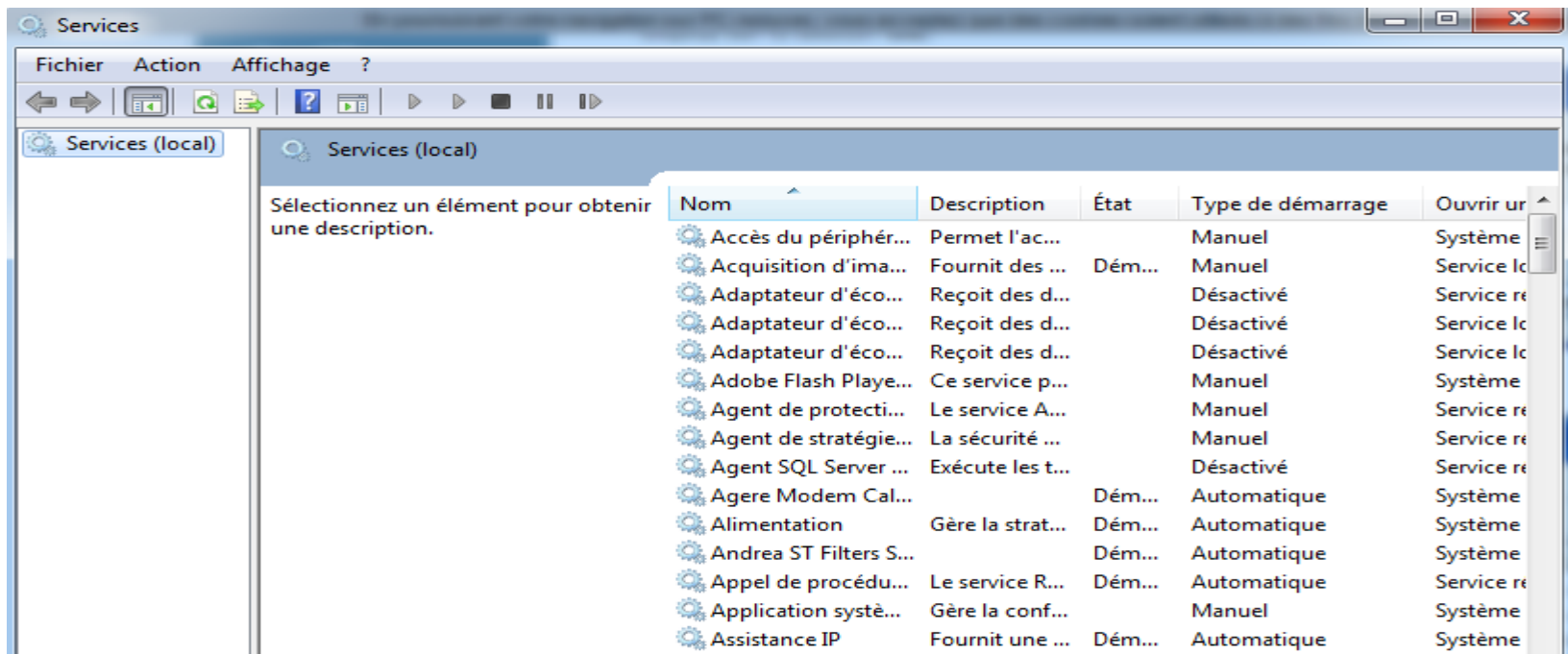
```
ServiceHost host = new  
ServiceHost(typeof(WcfService1.Service1));  
host.Open();
```

**-Pour arrêter le service:**

```
ServiceHost host = new  
ServiceHost(typeof(WcfService1.Service1));  
host.Close();
```

# Windows Service Hosting

-Contrôler le service à partir de windows.



**Remarque:** Pour accéder aux services aller à Exécuter et taper « services.msc »

# Windows Service Hosting

- Les étapes

1. Créer nouveau projet de type windows Service

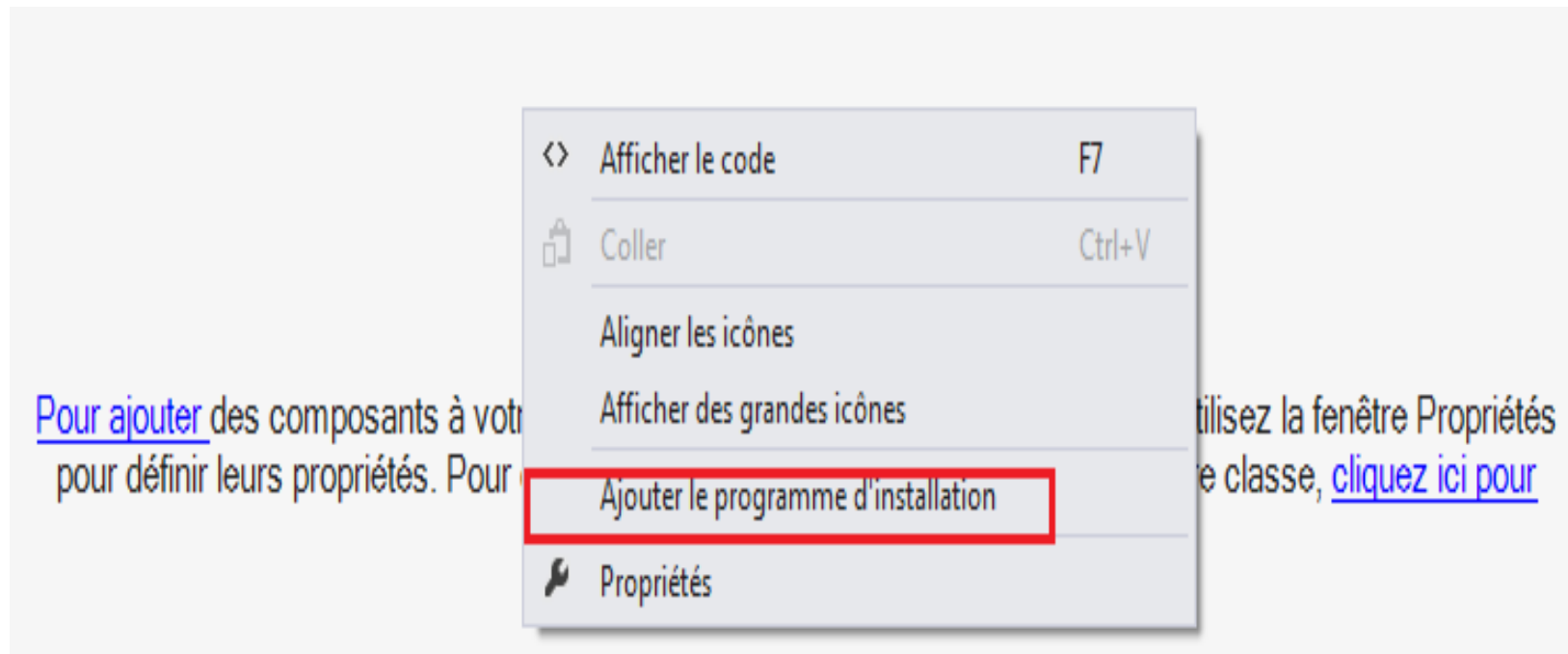
2.

Événement Onstart ()	Événement OnStop()
<b>host = new ServiceHost(typeof(WcfService1.S ervice1)); host.Open();</b>	<b>host.Close();</b>

3. Ajouter la même configuration de fichier App.config effectué dans le self hosting.

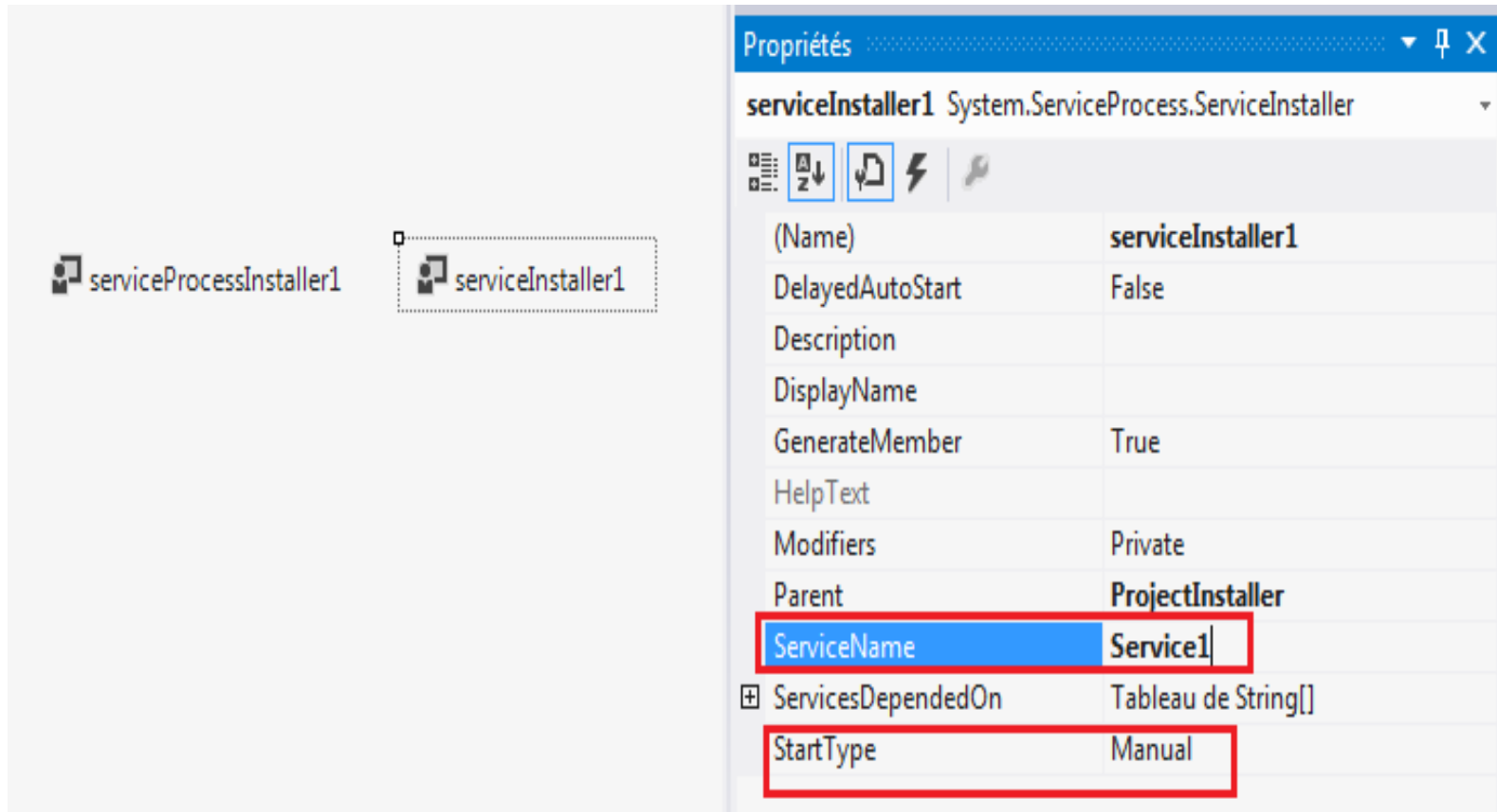
# Windows Service Hosting

4. Pour que ce service apparait comme un composant dans la liste des services windows.



# Windows Service Hosting

## 4.a. Configurer les propriétés de service Installer



The screenshot shows the Visual Studio IDE with a project named `serviceProcessInstaller1` containing a `serviceInstaller1` component. The Properties window is open, displaying the configuration for `serviceInstaller1` (System.ServiceProcess.ServiceInstaller).

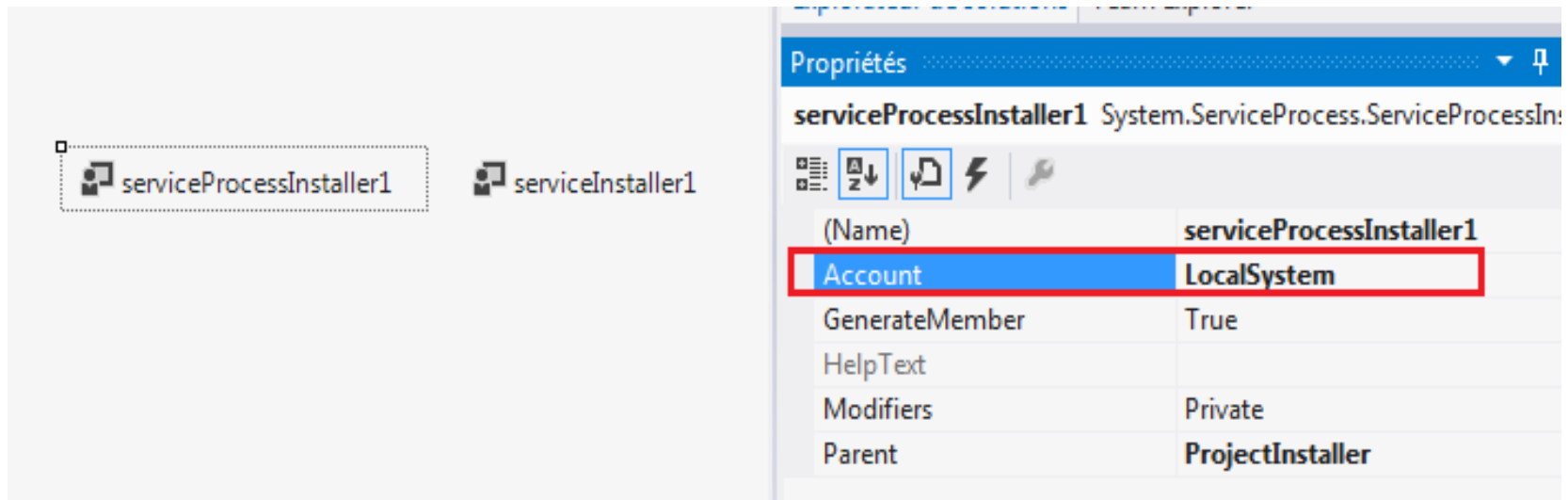
Property	Value
(Name)	serviceInstaller1
DelayedAutoStart	False
Description	
DisplayName	
GenerateMember	True
HelpText	
Modifiers	Private
Parent	ProjectInstaller
ServiceName	Service1
ServicesDependedOn	Tableau de String[]
StartType	Manual

The `ServiceName` and `StartType` properties are highlighted with red boxes.

# Windows Service Hosting

4.b. Configurer les propriétés de service Process Installer: Pour déterminer l'administrateur de service.

-localSystem signifie que n'importe quel utilisateur pourra contrôler le service





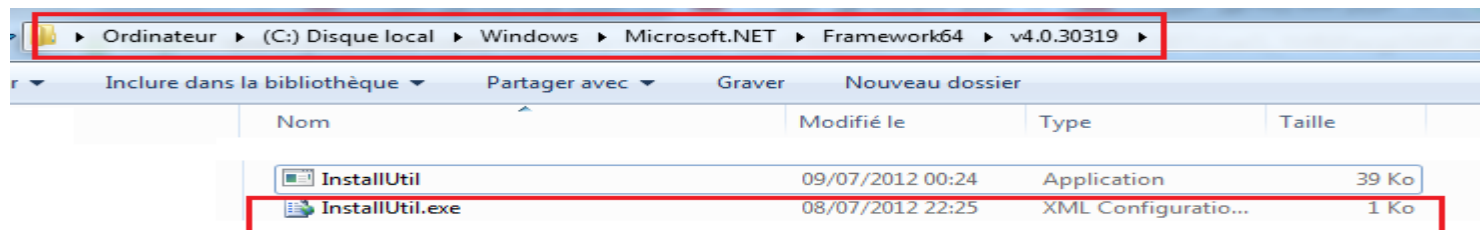
# Windows Service Hosting

## 5. Installation de service

- Les projets de type windows service ne s'exécute pas directement mais il faut les installer
- Pour cela on utilise l'utilitaire de ligne de commande **InstallUtil.exe**.

```
installutil -i yourproject.exe
```










**Remarque:** InstallUtil existe dans le chemin suivant



# Windows Service Hosting

## 6. Le service est bien ajoutée

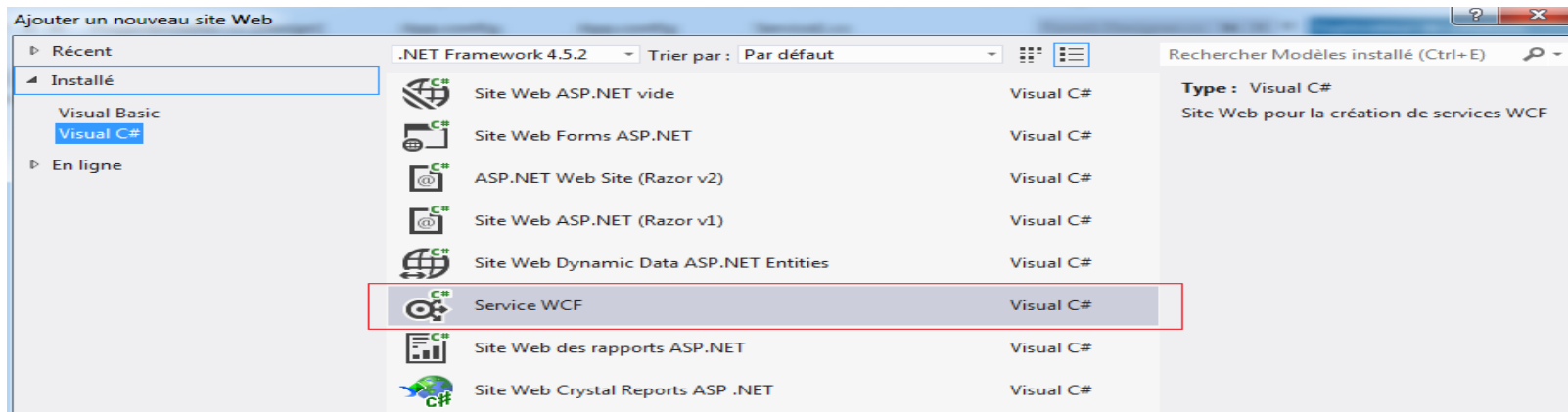
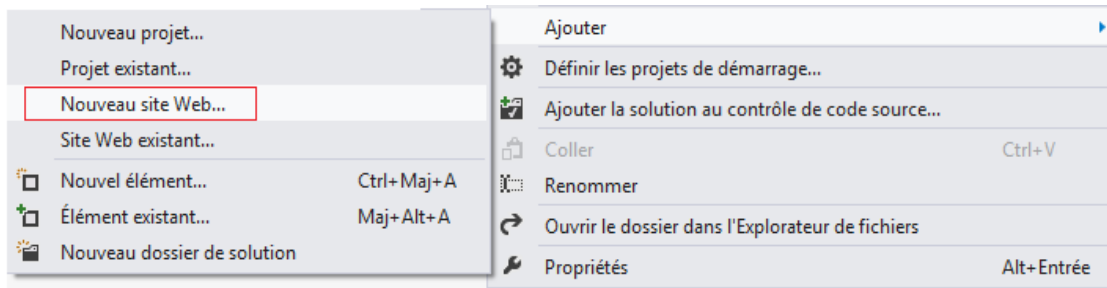
Démarrer le service

	Service SQL Active...	Permet l'int...	Désactivé	Service réseau
	Service SSTP (Sec...	Prend en ch...	Manuel	Service local
	Services Bureau à ...	Autorise les ...	Manuel	Service réseau
	Services de base d...	Active l'acc...	Manuel	Service local
	Services de chiffre...	Fournit qua...	Dém... Automatique	Service réseau
	Servicewcf1		Manuel	Système local
	Spouleur d'impres...	Charge les f...	Dém... Automatique	Système local
	SQL Server (SQLEX...	Permet de s...	Dém... Automatique	Service réseau
	SQL Server Browser	Fournit des ...	Dém... Automatique	Service local

# IIS Hosting

- Les étapes

1. Ajouter nouveau projet de type Web site (WCF service)



# IIS Hosting

- Les étapes

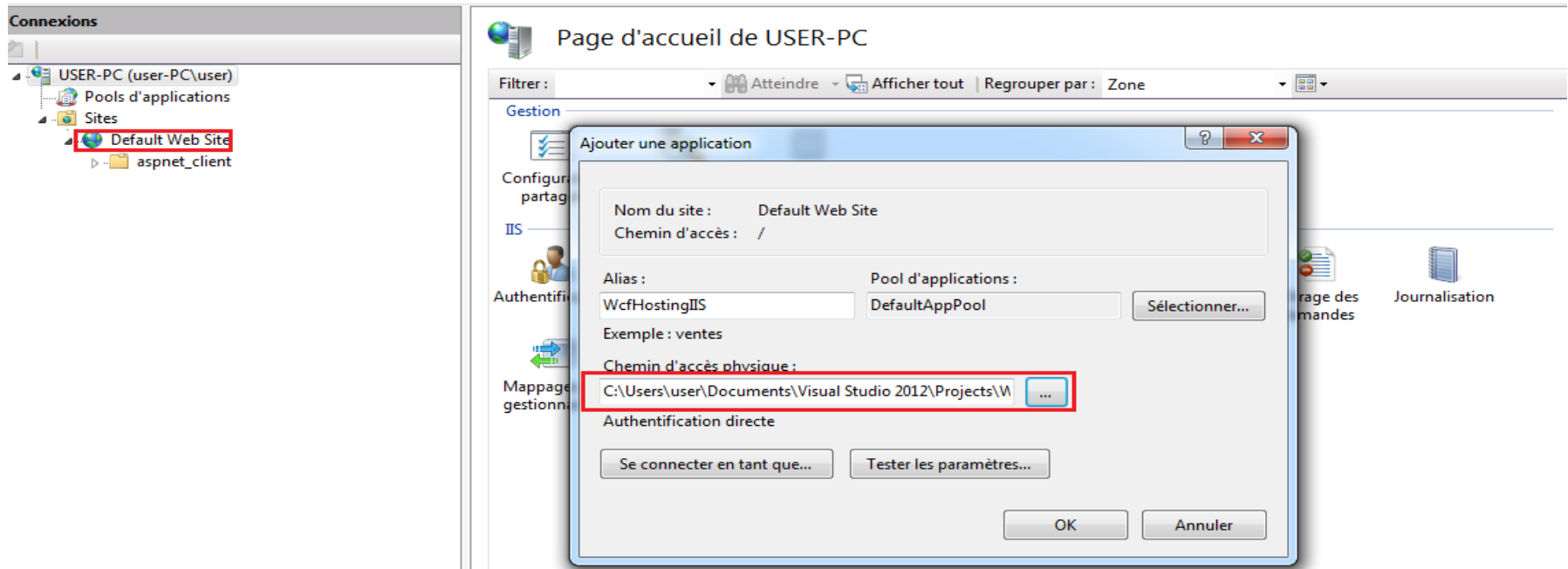
2. Ajouter la même configuration de fichier App.config effectué dans le self hosting.
3. Dans le fichier service.svc de ce projet , ajouter une référence du service wcf à héberger

```
<%@ ServiceHost Language="C#" Debug="true" Service="WcfService1.Service1" %>
```

# IIS Hosting

- Les étapes

4. En IIS ajouter une nouvelle application et référencer par ce projet



# IIS Hosting

- Les étapes

5. Afficher le fichier service.svc dans le navigateur

6. Par la suite via cette adresse qu'on pourra consommer le service



The screenshot shows a web browser window. The address bar contains the URL `localhost:4406/ServiceIIS.svc`, which is highlighted with a red box. Below the address bar, the page title is `Service Service1`. The main content area displays the text: "Vous avez créé un service." followed by a paragraph: "Pour tester ce service, vous allez devoir créer un client et l'utiliser pour appeler le service. Pour ce faire, vous pouvez utiliser l'outil svcutil.exe à partir de la ligne de commande avec la syntaxe suivante :". Below this text, there is a code block with the command: `svcutil.exe http://localhost:4406/ServiceIIS.svc?wsdl`. At the bottom, it says: "Il est également possible d'accéder à la description du service dans un fichier unique :", followed by another code block with the URL: `http://localhost:4406/ServiceIIS.svc?singleWsdl`.

← → ↻ ⓘ localhost:4406/ServiceIIS.svc

## Service Service1

Vous avez créé un service.

Pour tester ce service, vous allez devoir créer un client et l'utiliser pour appeler le service. Pour ce faire, vous pouvez utiliser l'outil svcutil.exe à partir de la ligne de commande avec la syntaxe suivante :

```
svcutil.exe http://localhost:4406/ServiceIIS.svc?wsdl
```

Il est également possible d'accéder à la description du service dans un fichier unique :

```
http://localhost:4406/ServiceIIS.svc?singleWsdl
```

# -WCF SOAP- est-il mort ?

- Non, ce n'est pas encore mort, il est sur son lit de mort mais quand?

*-Quand on veut créer un service Web basé sur HTTP, on devra utiliser API Rest au lieu de WCF car il est plus simple, plus léger et prend en charge les méthodologies de développement modernes dans lesquelles l'injection de dépendance est une exigence essentielle*

*Et alors?*

- *WCF est encore là si: on a besoin des communications non HTTP, telles que la messagerie unidirectionnelle, les files d'attente de messages, la communication duplex, etc, ou lorsque SOAP est une exigence*