

Bài 2

MÔ HÌNH LẬP TRÌNH VỚI WINDOWS COMMUNICATION FOUNDATION

Mục lục

1	Mô hình lập trình WCF	2
1.1	Sử dụng phương pháp hướng đối tượng hay hướng dịch vụ?	2
1.2	Service Model (Mô hình dịch vụ)	4
1.3	Các phương pháp lập trình với WCF	6
1.3.1	Declarative programming (Phương pháp khai báo)	6
1.3.2	Explicit programming (Phương pháp lập trình trực tiếp)	7
1.3.3	Phương pháp sử dụng tập tin cấu hình	7
2	Xây dựng một dịch vụ WCF	9
2.1	Cài đặt WCF	9
2.1.1	.NET Framework 3.5 SP1	9
2.1.2	Visual Studio 2008 SP1	9
2.2	Tạo dịch vụ WCF đầu tiên của bạn	9
2.2.1	Tạo ứng dụng phía server	9
2.2.2	Phát hành thông tin về dịch vụ	18
2.2.3	Tạo ứng dụng phía client	23
2.3	Cách khác để tạo tham chiếu ở client	25
3	Câu hỏi ôn tập	27
4	Tài liệu tham khảo	28

Nhìn lướt qua thì thấy rằng WCF có vẻ rối rắm phức tạp. Tuy nhiên nếu bạn hiểu về WSDL (Web Service Description Language) và về Web services (các dịch vụ web), thì việc nắm được các khái niệm trong WCF rất đơn giản. Nhưng nếu không quen về các khái niệm trên, để hiểu được khái niệm trong WCF thông qua bài trước quả là một việc hơi quá sức, do WCF có quá nhiều lớp và thành phần.

Tuy vậy, WCF hay ở chỗ là nó đem đến một mô hình lập trình và dịch vụ rất đơn giản. Mặc dù nền tảng bên dưới cho WCF tương đối lớn, WCF được xây dựng trên cơ sở .NET Framework do đó bạn có thể sử dụng ngôn ngữ lập trình và nền tảng quen thuộc để làm việc với nó. Thêm nữa, khi phát triển các dịch vụ với WCF bạn chỉ cần biết cách sử dụng một phần của các lớp đó mà thôi. Cách tốt nhất để học cách phát triển các dịch vụ với WCF là làm việc trực tiếp với nó. Bài này giới thiệu với các bạn mô hình lập trình với WCF, đồng thời hướng dẫn các bạn xây dựng một dịch vụ WCF đầu tiên.

1 Mô hình lập trình WCF

1.1 Sử dụng phương pháp hướng đối tượng hay hướng dịch vụ?

Nếu các bạn đã học qua môn học về các phương pháp lập trình hẳn sẽ thấy có 2 phương pháp chính là hướng thủ tục và hướng đối tượng. Và phương pháp hướng đối tượng trong thời gian gần đây được phát triển rất mạnh và được hỗ trợ ngay trong các ngôn ngữ lập trình như C# hay VB.NET. Khi làm việc với các dịch vụ web bạn đã làm quen với một phương pháp nữa là lập trình hướng dịch vụ (Service-oriented programming). Microsoft đã cung cấp nhiều công cụ trên .NET Framework để hỗ trợ phương pháp lập trình này thông qua các lớp trong không gian tên: System.Web.Services.

Như vậy khi làm việc với WCF bạn có hai lựa chọn là hướng đối tượng và hướng dịch vụ. Vậy ta nên sử dụng phương pháp nào? Câu trả lời là dùng cả hai. Nói một cách đơn giản là phương pháp hướng đối tượng được sử dụng để phát triển các ứng dụng trên desktop, còn phương pháp hướng dịch vụ được sử dụng để kết nối các ứng dụng đó với nhau. Điều quan trọng ở đây là làm sao để hiểu được sự khác nhau giữa hai phương pháp và hiểu được khi nào chúng được sử dụng và sử dụng như thế nào đồng thời cũng phải hiểu về các lợi ích chúng cung cấp.

Về hướng đối tượng có thể hiểu như sau. Các ứng dụng hướng đối tượng là hai hay nhiều lớp phụ thuộc lẫn nhau và chia sẻ chung các kiểu dữ liệu. Những lớp này liên lạc với nhau thông qua các lời gọi các hàm mà lớp đối tượng cung cấp.

Các ứng dụng hướng dịch vụ là các chương trình không biết gì về nhau. Mỗi ứng dụng liên lạc với ứng dụng khác thông qua các bản tin. Điểm đặc biệt là các bản tin này được gửi từ một ứng dụng sang ứng dụng khác mà không quan tâm tới nền tảng mà dịch vụ đang chạy.

Khi phát triển các dịch vụ WCF, điều quan trọng là cần hiểu sự liên kết giữa hướng đối tượng và hướng dịch vụ. Khi làm việc với .NET Framework bạn chắc chắn rất quen thuộc với thuật ngữ lớp (class)

và giao diện (interface). Các thuật ngữ này vẫn được sử dụng khi phát triển dịch vụ WCF. Các lớp và giao diện là phần hướng đối tượng trong WCF, còn phần hướng dịch vụ trong WCF sẽ được thấy khi bạn đưa vào các thuộc tính WCF để định nghĩa các thực thể.

Ví dụ, lớp sau đây định nghĩa một giao diện hướng đối tượng

```
public interface DịchVuBanHang
{
    decimal TinhGiaVanChuyen(string diachi, decimal trongluong)
    {
        // thực hiện tính toán
    }

    decimal TinhTienThue(decimal tongGiaTien)
    {
        // thực hiện tính toán
    }
}
```

Ta sẽ có phần hướng dịch vụ cho dịch vụ WCF khi thêm vào các thuộc tính cho giao diện ở trên

```
[ServiceContract]
public interface DịchVuBanHang
{
    [OperationContract]
    decimal TinhGiaVanChuyen(string diachi, decimal trongluong)
    {
        // thực hiện tính toán
    }

    [OperationContract]
    decimal TinhTienThue(decimal tongGiaTien)
    {
        // thực hiện tính toán
    }
}
```

Như vậy là qua ví dụ trên các bạn có thể thấy mối liên kết giữa phương pháp hướng đối tượng và hướng dịch vụ trong WCF. Các bạn chưa cần quan tâm tới các thuộc tính [ServiceContract] và [OperationContract] vội, bởi vì những thuộc tính này, và còn nhiều thứ khác nữa sẽ được giới thiệu một cách chi tiết khi thích hợp.

1.2 Service Model (Mô hình dịch vụ)

Nếu bạn đã từng làm việc với dịch vụ web, bạn sẽ thấy mô hình này quen thuộc với bạn theo một cách nào đó. Khi bạn tạo một dịch vụ web, bạn thực sự tạo ra một dịch vụ (service). Dịch vụ web chứa một tài liệu XML để mô tả tất cả mọi thứ cần biết về dịch vụ đó. Tài liệu này được mô tả bằng ngôn ngữ Web Service Description Language (ngôn ngữ mô tả dịch vụ web). Nó chứa ba phần:

- **Service (dịch vụ):** Chứa thông tin về vị trí của dịch vụ
- **Binding:** Chứa thông tin về cách liên lạc với dịch vụ, như dịch vụ sử dụng giao thức gì, vv.
- **PortType (kiểu cổng):** Giải thích về dịch vụ sẽ làm gì

Mô hình dịch vụ trên WCF cũng tương tự như với mô hình dịch vụ web. Điểm khác biệt là ở cách đặt tên. Trong WCF các phần không được gọi là service, binding, và portType mà được gọi tương ứng là address (địa chỉ), binding, và contract.

Mô hình dịch vụ WCF được cung cấp trong không gian tên System.ServiceModel. Không gian tên này chứa rất nhiều lớp, nhưng bạn hoàn toàn không cần biết toàn bộ chúng. Để sử dụng mô hình và xây dựng dịch vụ, ta thường sử dụng một số lớp sau:

Lớp	Mô tả
BasicHTTPBinding	Là binding mà các điểm cuối dịch vụ có thể sử dụng để liên lạc với các ứng dụng khách và dịch vụ web (ASMX)
NetMsmqBinding	Là binding mà các điểm cuối dịch vụ có thể sử dụng để liên lạc với các MSMQ khách và các dịch vụ khác
NetNamedPipeBinding	Là binding mà các điểm cuối dịch vụ có thể sử dụng để liên lạc với các ứng dụng khách/dịch vụ trên cùng một máy
NetTCPBinding	Là binding mà các điểm cuối dịch vụ có thể sử dụng để liên lạc với các ứng dụng khách/dịch vụ ở các máy khác nhau
WSHTTPBinding	Là binding mà các điểm cuối dịch vụ có thể sử dụng để liên lạc với các ứng dụng khách/dịch vụ sử dụng các giao dịch phân tán và các phiên làm việc bảo mật và tin cậy được.
EndpointAddress	Lớp biểu diễn địa chỉ duy nhất được cung cấp và truy xuất được

	cho máy khách để liên lạc với điểm cuối dịch vụ
EndpointAddressBuilding	Là phương pháp để tạo mới các địa chỉ đầu cuối với các giá trị tham số xác định
ChannelFactory	Là phương pháp trong đó các kiểu kênh khác nhau được tạo ra và quản lý, và đưa tới cho các ứng dụng khách để gửi bản tin tới các điểm cuối
Identity	Cách mà một định danh được xác định, cho phép xác thực giữa các điểm cuối khi trao đổi bản tin
MessageHeader	Biểu diễn nội dung của một đầu đề bản tin SOAP
ServiceHost	Phương pháp cung cấp vật chứa cho các dịch vụ
ReliableSession	Cung cấp truy xuất tới các thuộc tính của thành phần binding trong phiên làm việc tin cậy.

Để định nghĩa việc liên lạc của dịch vụ, ta thường hay sử dụng các lớp sau

Lớp	Mô tả
AddressHeader	Phần đầu đề chứa thông tin địa chỉ được sử dụng để xác định và liên lạc với một điểm cuối
AddressHeaderCollection	Một tập hợp các đầu đề địa chỉ
Binding	Tập hợp các thành phần binding, mỗi binding định nghĩa cách mà một điểm cuối liên lạc với thế giới bên ngoài
BindingContext	Cung cấp địa chỉ và thông tin binding cần thiết cho việc xây dựng kênh
BindingElement	Biểu diễn một thành phần binding, được sử dụng để xây dựng các binding
CustomBinding	Sử dụng để định nghĩa và xây dựng một tùy biến binding từ một tập các thành phần binding
Message	Một đơn vị của liên lạc giữa các điểm cuối

MessageHeader	Nội dung của đầu đề bản tin SOAP
MessageHeaders	Tập hợp các đầu đề bản tin

1.3 Các phương pháp lập trình với WCF

Có một số phương pháp lập trình với WCF, mỗi phương pháp có ưu điểm và khuyết điểm riêng của nó. Điều đặc biệt về WCF là luôn có hơn một cách để giải quyết một vấn đề trong WCF, và bạn không nhất thiết phải chọn duy nhất một phương pháp nào. Trong thực tế, cách làm tốt nhất là tổ hợp các phương pháp để có được sự linh hoạt và mềm dẻo cho dịch vụ của bạn.

Có ba phương pháp hay được sử dụng khi phát triển dịch vụ WCF như sau:

- Phương pháp khai báo
- Phương pháp lập trình trực tiếp
- Phương pháp sử dụng tập tin cấu hình

1.3.1 Declarative programming (Phương pháp khai báo)

Lập trình khai báo đạt được thông qua các thuộc tính. Những thuộc tính này được sử dụng để định nghĩa các contract và xác định hành xử của dịch vụ. Chúng được sử dụng để xác định thêm các tham số để thay đổi các chi tiết của contract và hành xử dịch vụ.

Thuộc tính `ServiceContract` dùng để quy định là giao diện này định nghĩa các chức năng của một dịch vụ. Thuộc tính `OperationContract` được sử dụng ở các hàm để quy định rằng hàm này được khai báo là một phần của dịch vụ. Đó là tất cả những gì cần để tạo ra một dịch vụ WCF.

Thêm nữa, bạn không nhất thiết phải sử dụng các giao diện (interface) khi cài đặt một dịch vụ, điều này cũng giống như việc bạn không cần phải sử dụng giao diện để định nghĩa một lớp. Tuy vậy bạn nhất thiết phải quy định phần nào thuộc về dịch vụ. Bạn có thể định nghĩa những phần khác cần cho giao diện, nhưng chỉ những hàm (phương thức) có gắn thuộc tính `[OperationContract]`.

Ví dụ ta có một dịch vụ thực hiện phép tính cộng giữa 2 số nguyên `AddInt` và 2 số thực `AddDouble`. Ta khai báo dịch vụ như sau:

```
[ServiceContract]
public interface ICalcService
{
    [OperationContract]
    int AddInt(int x, int y);
}
```

```
[OperationContract]
double AddDouble(double x, double y);
}
```

Như vậy dịch vụ của chúng ta sau khi khai báo sẽ có 2 phương thức (khai báo với thuộc tính `OperationContract`) là `AddInt` và `AddDouble`. Tuy nhiên khi khai báo trong C#, việc đặt tên `AddInt` và `AddDouble`, và có thể có một số hàm add cho các kiểu dữ liệu khác, có thể rút gọn lại thành một tên hàm `Add` mà thôi. Nhưng các dịch vụ lại không cho phép đặt trùng tên hàm như thế. Chúng ta có thể khai báo thêm với thuộc tính `OperationContract` để thực hiện, cách làm như sau:

```
[ServiceContract]
public interface ICalcService
{
    [OperationContract(Name="AddInt")]
    int Add(int x, int y);

    [OperationContract(Name="AddDouble")]
    double Add(double x, double y);
}
```

Các bạn có thể thấy là chúng ta sử dụng được phép nạp chồng tên trong C# và sử dụng thêm tham số `Name` để quy định thêm tên hàm ở dịch vụ. Ngoài ưu điểm trong việc giải quyết nạp chồng tên hàm, ta còn thấy một lợi ích khác nữa là, việc quy định tham số `Name` trong thuộc tính `OperationContract` còn cho ta thêm linh hoạt trong việc đổi tên các hàm trong giao diện mà không làm thay đổi định nghĩa dịch vụ, nghĩa là các ứng dụng khác sử dụng dịch vụ này không cần phải biên dịch lại.

1.3.2 Explicit programming (Phương pháp lập trình trực tiếp)

Là phương pháp lập trình hướng đối tượng, bạn làm việc trực tiếp với các lớp và giao diện cung cấp bởi mô hình đối tượng của WCF. Làm việc trực tiếp với mô hình đối tượng cho phép nhà phát triển tính linh hoạt cao hơn và khả năng điều khiển tốt hơn thông qua mã nguồn của họ. Thêm nữa nó cho phép điều khiển sâu hơn rất nhiều so với phương pháp khai báo và phương pháp sử dụng tập tin cấu hình.

1.3.3 Phương pháp sử dụng tập tin cấu hình

Cũng giống như phương pháp khai báo, có rất nhiều thứ mà bạn có thể quy định liên quan đến hành xử của một dịch vụ thông qua tập tin cấu hình của dịch vụ. Điều hay trong cách tiếp cận này là những thay đổi ở tập tin cấu hình hoàn toàn không cần phải biên dịch lại dịch vụ mới sử dụng được.

Sau đây là ví dụ sử dụng tập tin cấu hình để định nghĩa dịch vụ tính toán trong ví dụ của phần phương pháp khai báo.

```
<?xml version="1.0" encoding="utf-8" ?>
```

```

<configuration>
  <system.web>
    <compilation debug="true" />
  </system.web>
  <system.serviceModel>
    <services>
      <service
behaviorConfiguration="CalculationService.CalcServiceBehavior"
        name="CalculationService.CalcService">
        <endpoint address="" binding="wsHttpBinding"
contract="CalculationService.ICalcService">
          <identity>
            <dns value="localhost" />
          </identity>
        </endpoint>
        <endpoint address="mex" binding="mexHttpBinding"
contract="IMetadataExchange" />
        <host>
          <baseAddresses>
            <add baseAddress="http://localhost:8731/CalcService/" />
          </baseAddresses>
        </host>
      </service>
    </services>
    <behaviors>
      <serviceBehaviors>
        <behavior name="CalculationService.CalcServiceBehavior">
          <serviceMetadata httpGetEnabled="true" />
          <serviceDebug includeExceptionDetailInFaults="false" />
        </behavior>
      </serviceBehaviors>
    </behaviors>
  </system.serviceModel>
</configuration>

```


2 Xây dựng một dịch vụ WCF

2.1 Cài đặt WCF

2.1.1 .NET Framework 3.5 SP1

Để xây dựng một dịch vụ WCF, đầu tiên bạn cần phải cài đặt .NET Framework 3.5 SP1. Thực ra chỉ cần .NET Framework 3.0 là đủ, tuy nhiên .NET Framework 3.5 SP1 còn cung cấp thêm cho bạn nhiều tính năng nữa, nên bạn nên cài .NET Framework 3.5 SP1. Bản cài đặt của framework được Microsoft cung cấp ở trang web <http://www.microsoft.com/downloads/details.aspx?familyid=ab99342f-5d1a-413d-8319-81da479ab0d7&displaylang=en> hoặc tại <http://download.microsoft.com/download/2/0/e/20e90413-712f-438c-988e-fdaa79a8ac3d/dotnetfx35.exe>

2.1.2 Visual Studio 2008 SP1

Sau khi cài đặt .NET Framework 3.5 SP1, bạn thực hiện cài đặt Visual Studio 2008 bản Express hoặc bản Professional tùy theo bạn có bản nào. Nếu kinh phí hạn hẹp, bạn có thể tải về bản Visual Studio 2008 Express Edition miễn phí trên trang web của Microsoft, link ở đây: <http://go.microsoft.com/?linkid=9350817>

Giờ đây bạn đã sẵn sàng để tạo ra dịch vụ đầu tiên trên WCF

2.2 Tạo dịch vụ WCF đầu tiên của bạn

Bạn có thể hình dung ứng dụng chúng ta sẽ xây dựng như sau. Công ty Contoso cần xây dựng một hệ thống quản lý các nhân viên của công ty. Ban đầu, chúng ta cần phải xây dựng một ứng dụng ở server

- Cung cấp danh sách các nhân viên, và
- Cho phép hỏi về ngày sinh của một nhân viên nào đó.

Sau đó cần một ứng dụng phía client để làm những việc sau

- Hiện thị danh sách các nhân viên
- Chọn một nhân viên và hiển thị ngày sinh của nhân viên đó.

2.2.1 Tạo ứng dụng phía server

Bước 1. Tạo ứng dụng

1. Mở Visual Studio 2008, chọn tạo mới C# Console Project đặt tên là StaffService, xem Figure 1

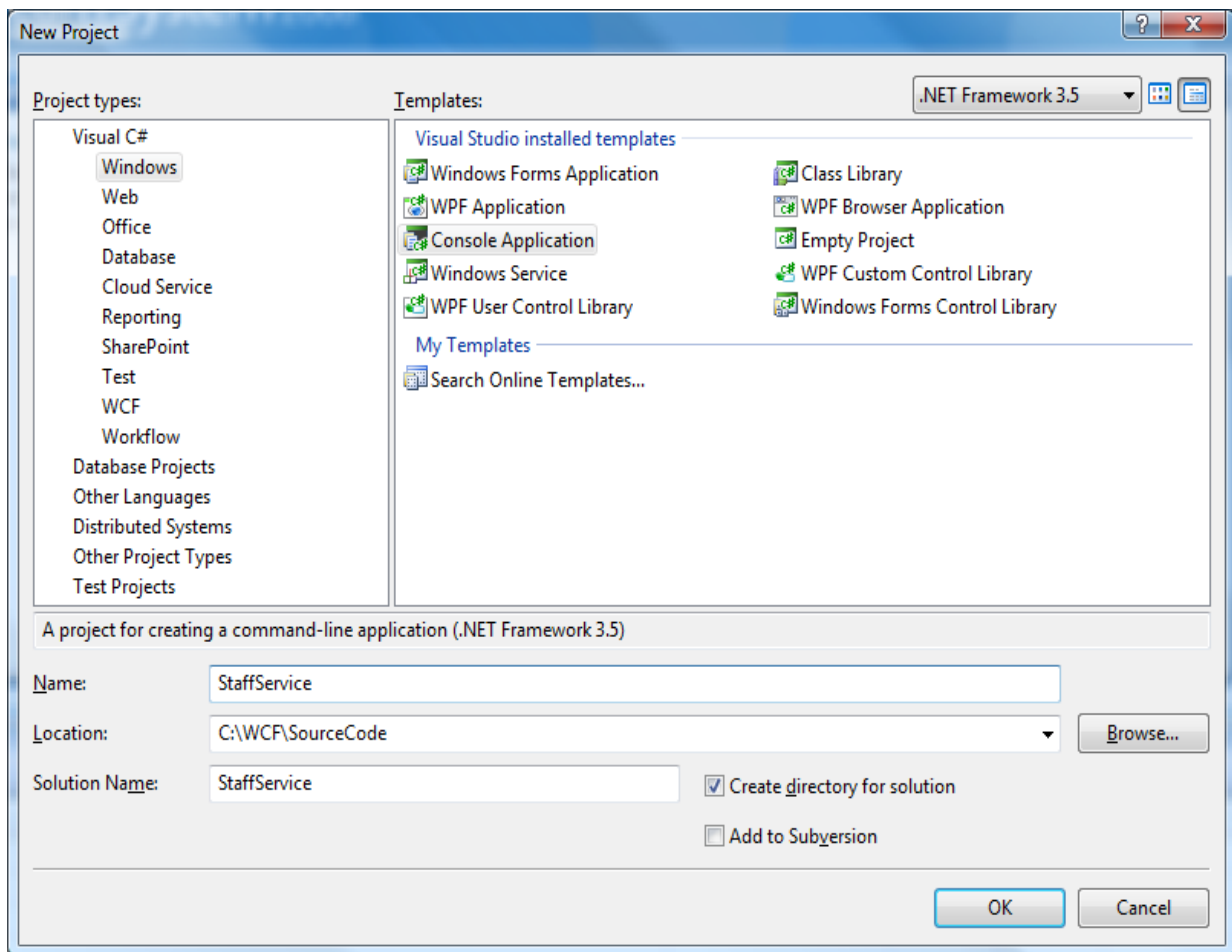


Figure 1 Tạo mới project

2. Thêm tham chiếu tới System.ServiceModel.dll.

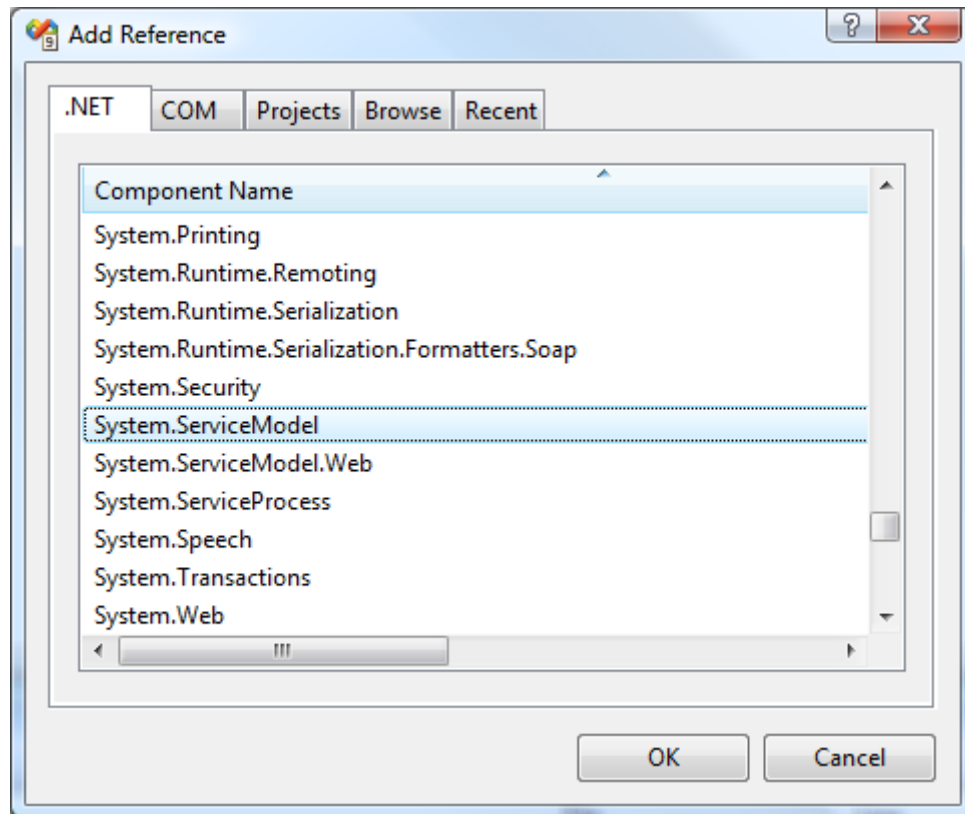


Figure 2 Tham chiếu tới ServiceModel

Bước 2. Tạo contract dịch vụ

1. Định nghĩa giao diện sẽ được sử dụng như là contract cho ứng dụng của chúng ta, thêm vào thuộc tính `ServiceContract` cho giao diện đó
2. Định nghĩa 2 hàm như đoạn mã nguồn sau, các hàm này đóng vai trò là các contract operations miêu tả chức năng của dịch vụ

```
[ServiceContract]
public interface IStaff
{
    [OperationContract]
    string DisplayStaff();
    [OperationContract]
    DateTime GetBirthday(int staffId);
}
```

Bước 3. Cài đặt dịch vụ

1. Định nghĩa lớp Staff để cài đặt giao diện IStaff
2. Thực hiện cài đặt cho 2 hàm được định nghĩa trong giao diện IStaff

```
using System;
```

```

namespace StaffService
{
    public class Staff : IStaff
    {
        #region IStaff Members

        public string DisplayStaff()
        {
            return "1. Lê Anh\n2. Trần Văn Bình\n3. Nguyễn Văn  
Cương\n4. Đinh Văn Dũng";
        }

        public DateTime GetBirthday(int staffId)
        {
            switch (staffId)
            {
                case 1:
                    return new DateTime(1979, 1, 20);
                case 2:
                    return new DateTime(1975, 5, 1);
                case 3:
                    return new DateTime(1967, 2, 26);
                case 4:
                    return new DateTime(1958, 10, 11);
                default:
                    return DateTime.Now;
            }
        }
    }
    #endregion
}

```

Bước 4. Tạo vật chứa dịch vụ

1. Thêm đoạn mã nguồn sau vào hàm main

```

ServiceHost sh = new ServiceHost(typeof(Staff));
try
{
    sh.Open();
    Console.WriteLine("Staff Service opened successfully");
    Console.WriteLine("Press Enter to terminate Staff Service");
}

```

```
        Console.ReadLine();  
    }  
    finally  
    {  
        sh.Close();  
    }  
}
```

Bước 5. Tạo các cấu hình dịch vụ

1. Compile project của bạn, đảm bảo rằng không có lỗi xảy ra trong quá trình biên dịch
2. Mở trình soạn thảo dịch vụ, Service Configuration Editor và nạp tệp ứng dụng, bằng cách trong Visual Studio 2008, chọn menu Tools, chọn lựa chọn WCF Service Configuration Editor
3. Chọn New Config, đặt tên là Staff.Service

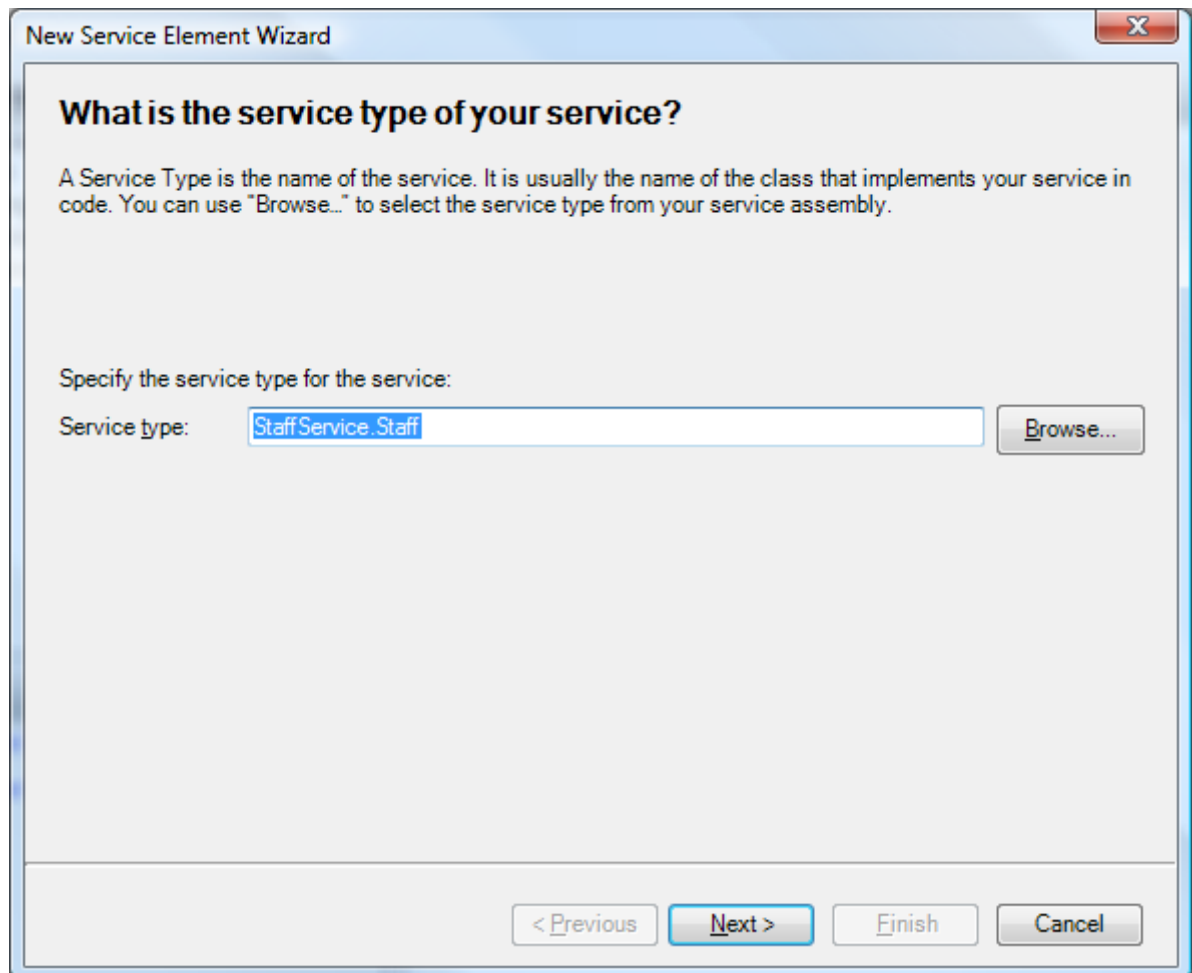


Figure 3 Tạo config cho service

4. Đặt contract dịch vụ là `StaffService.IStaff`

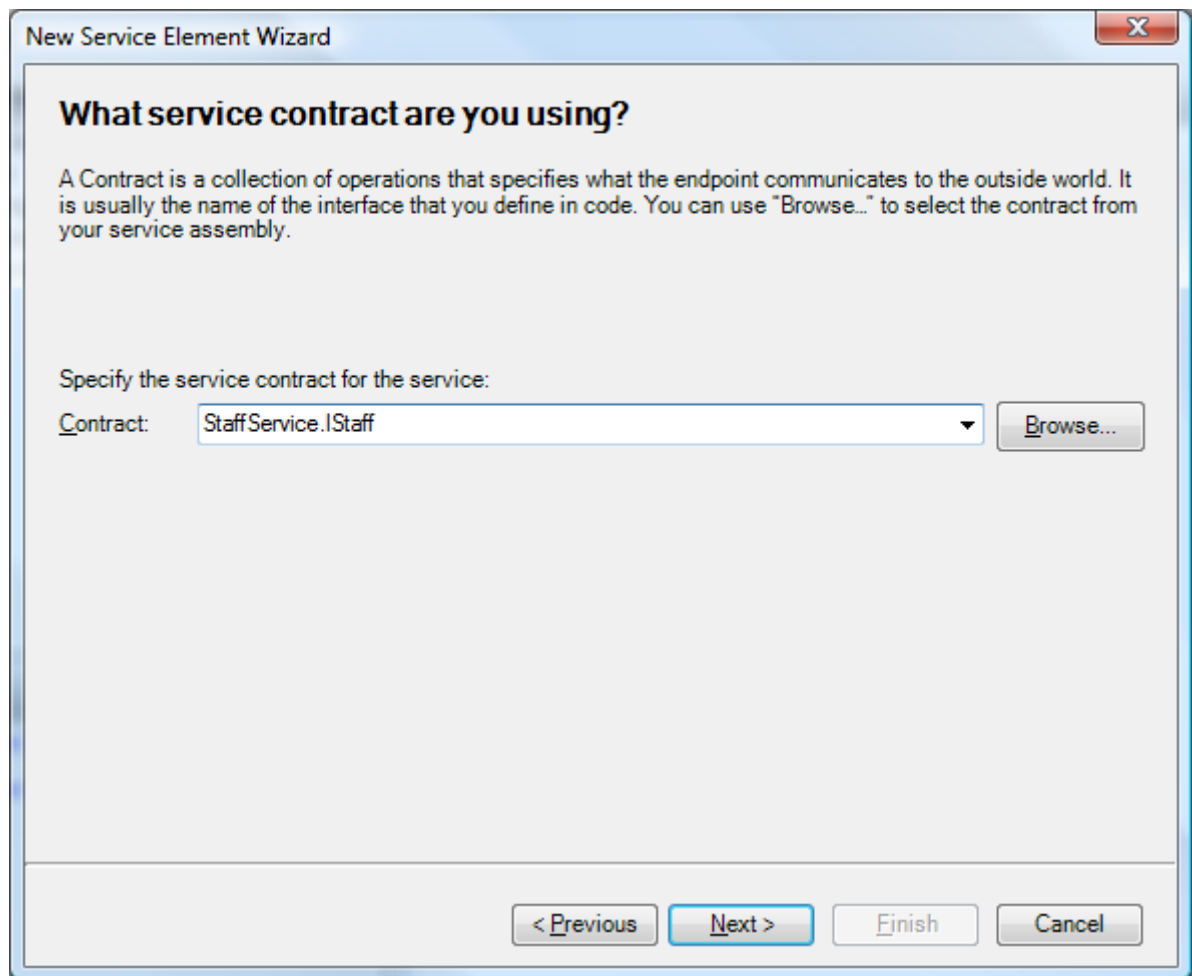


Figure 4 Đặt tên cho contract dịch vụ

5. Chọn cách liên lạc là HTTP

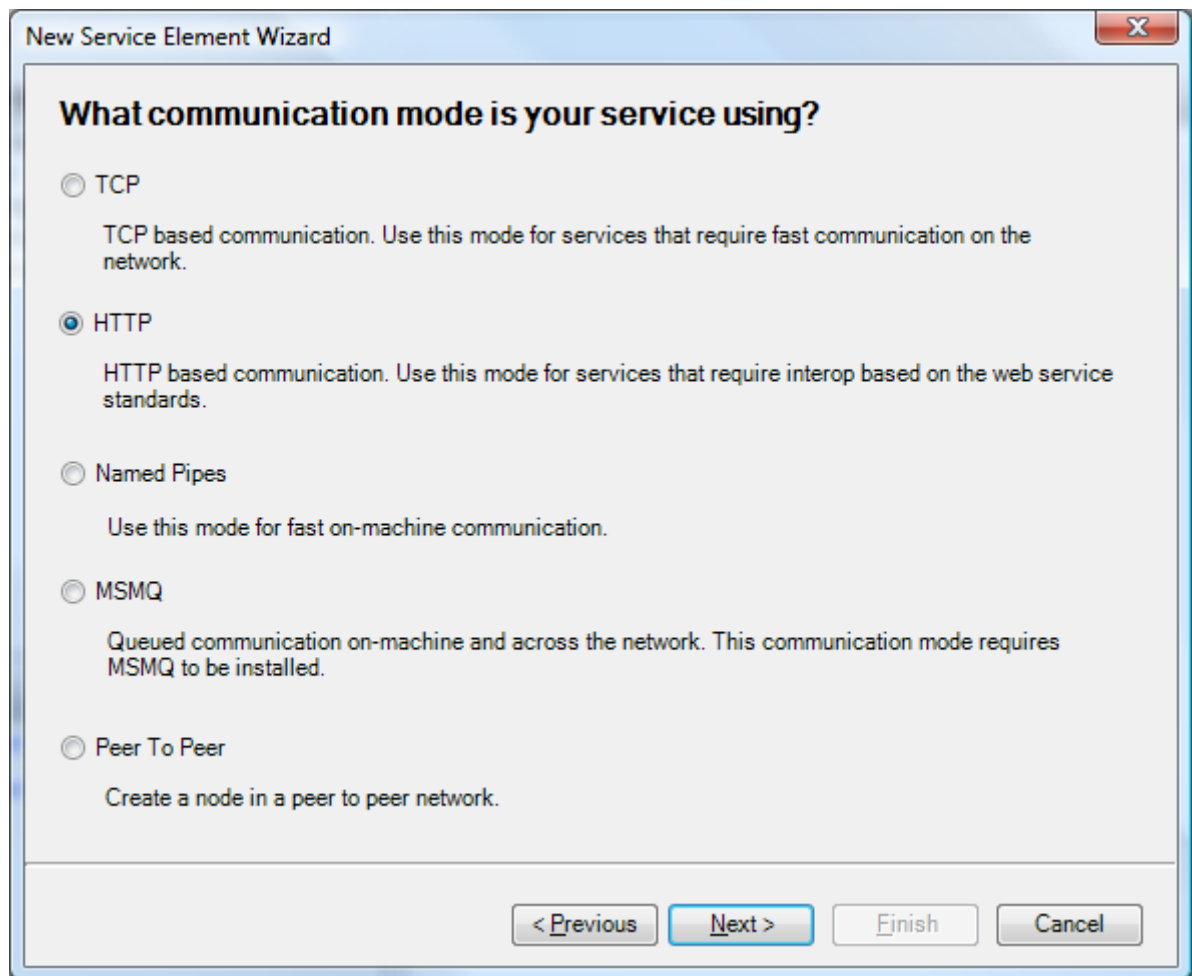


Figure 5 Chọn cách liên lạc

6. Tiếp theo chọn phương pháp làm việc là Advanced Web Services interoperability, và đặt kiểu liên kết là Simplex Communication

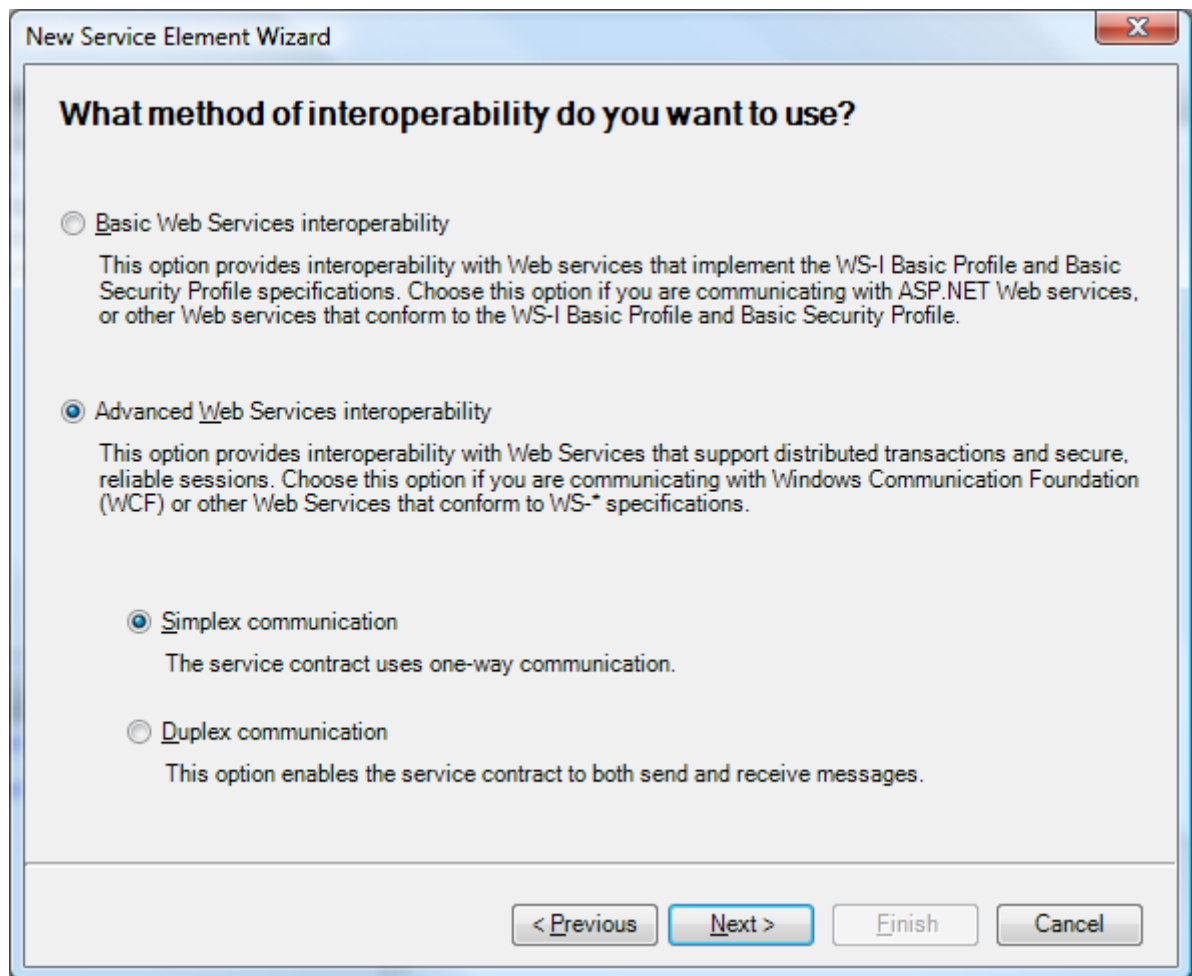


Figure 6 Chọn Simplex communication

7. Đặt mặc định (trống) cho trường địa chỉ, và bấm Finish, bạn sẽ thấy kết quả sau

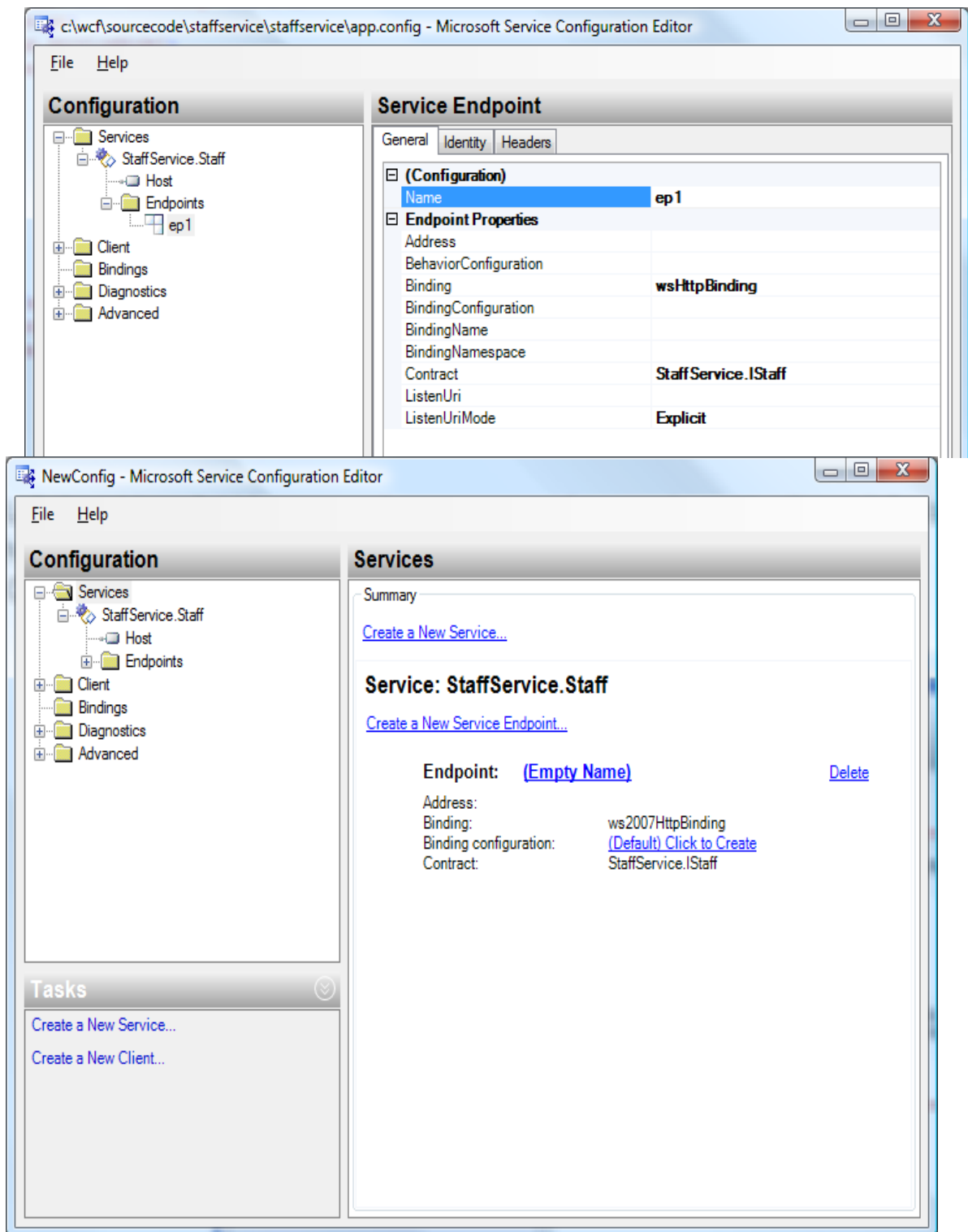


Figure 7 Kết quả cấu hình dịch vụ

8. Giờ bạn bấm vào biểu tượng Host phía bên trái, sau đó chọn thêm mới địa chỉ cơ sở cho dịch vụ của bạn, đặt địa chỉ là <http://localhost:8000/StaffService>

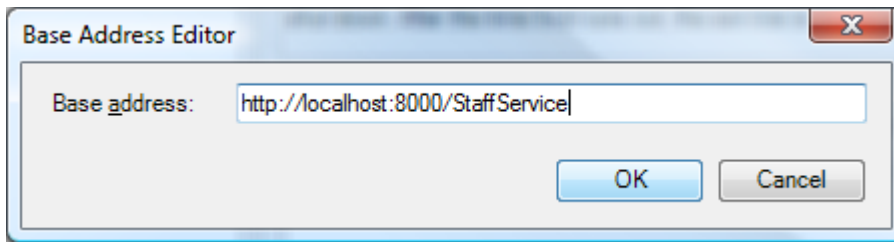


Figure 8 Thêm địa chỉ cơ sở

9. Mở rộng nút EndPoints, chọn điểm cuối, rồi đặt tên là ep1

Figure 9 Đặt tên cho điểm cuối

10. Kết thúc quá trình này bạn sẽ có tệp cấu hình như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <system.serviceModel>
    <services>
      <service name="StaffService.Staff">
        <endpoint address="" binding="wsHttpBinding"
bindingConfiguration=""
name="ep1" contract="StaffService.IStaff" />
        <host>
          <baseAddresses>
            <add
baseAddress="http://localhost:8000/StaffService" />
          </baseAddresses>
        </host>
      </service>
    </services>
  </system.serviceModel>
</configuration>
```

2.2.2 Phát hành thông tin về dịch vụ

Để phát hành thông tin về dịch vụ cho các ứng dụng khác khai thác, ta cần theo các bước sau:

1. Mở rộng nút Advanced (bên trái), sau đó chọn nút Service Behaviors và bấm vào New Service Behavior Configuration ở bên phải

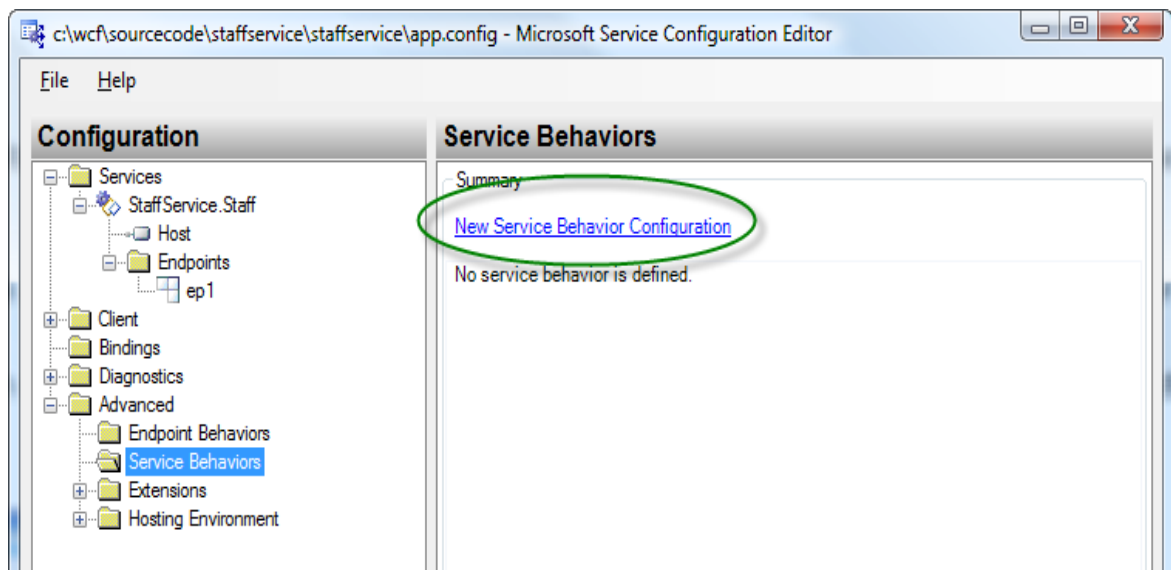


Figure 10 Tạo mới cấu hình cho hành xử dịch vụ

2. Đặt tên là MetadataBehavior, bấm Add để thêm thành phần hành xử, chọn ServiceMetadata

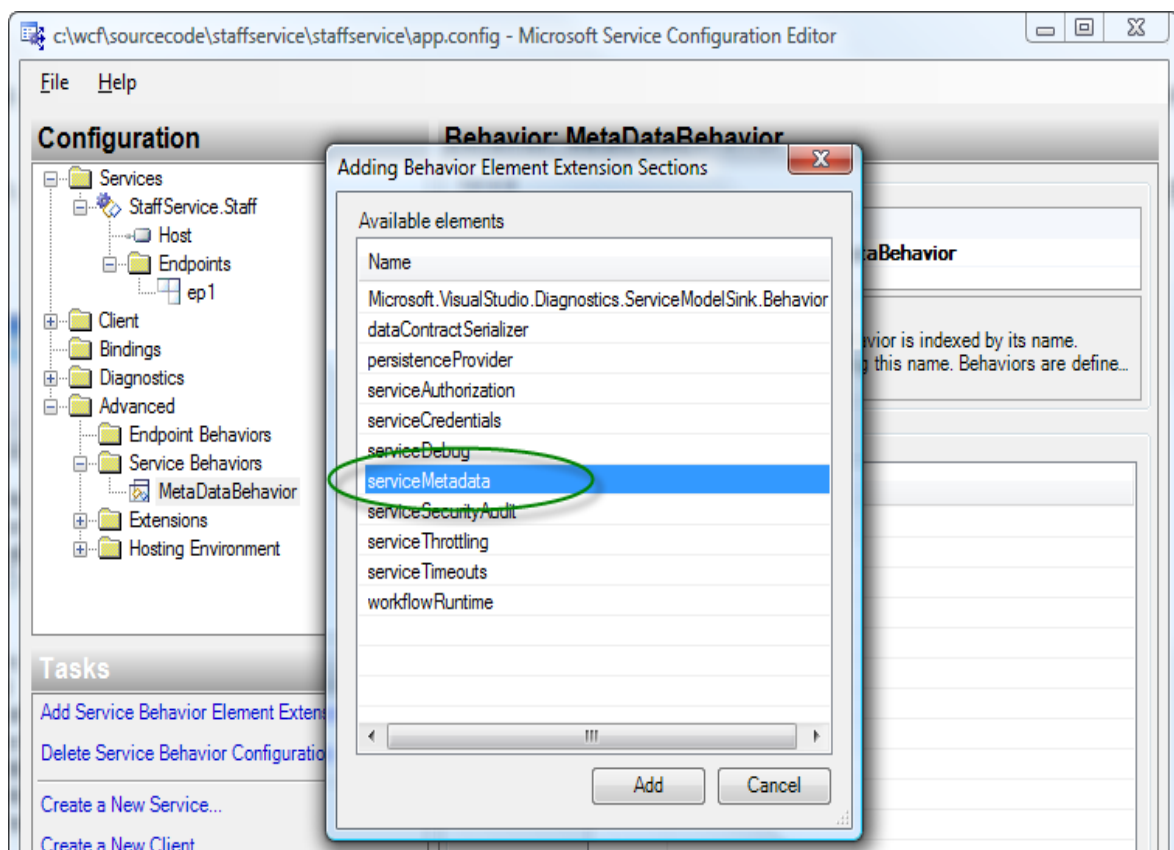


Figure 11 Thêm hành xử siêu dữ liệu

3. Chọn nút serviceMetadata ở bên trái và nhấp vào giá trị HttpGetEnabled thành true là HttpGetUrl là http://localhost:8000/StaffService

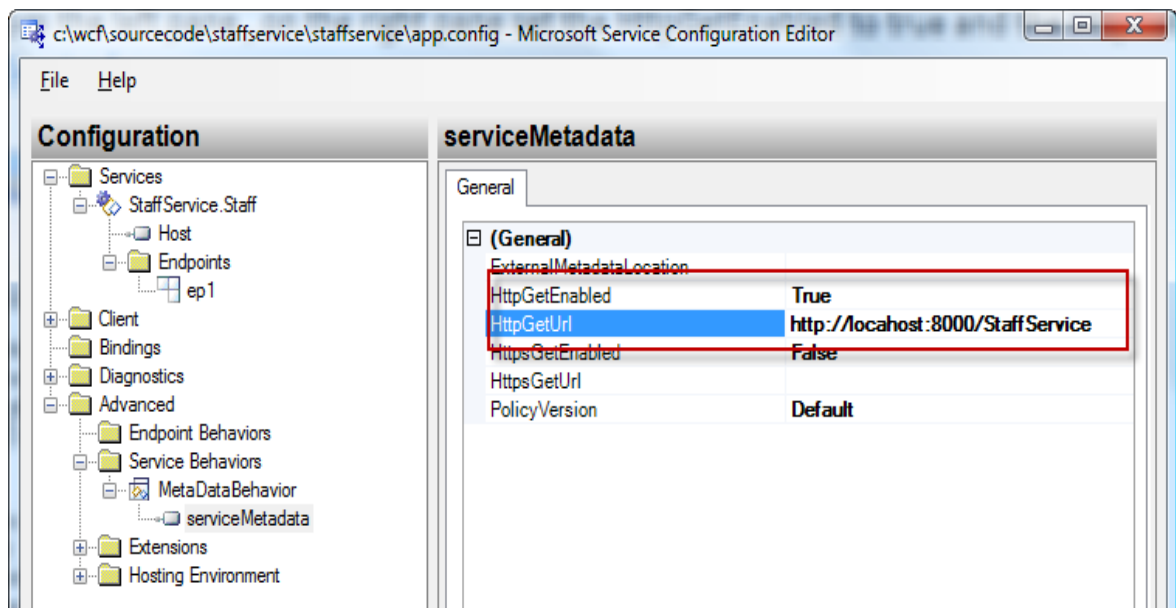


Figure 12 Đặt cấu hình cho serviceMetadata

4. Giờ bạn chọn nút StaffService.Staff và đặt Behavior Configuration với giá trị là “MetaDataBehavior”

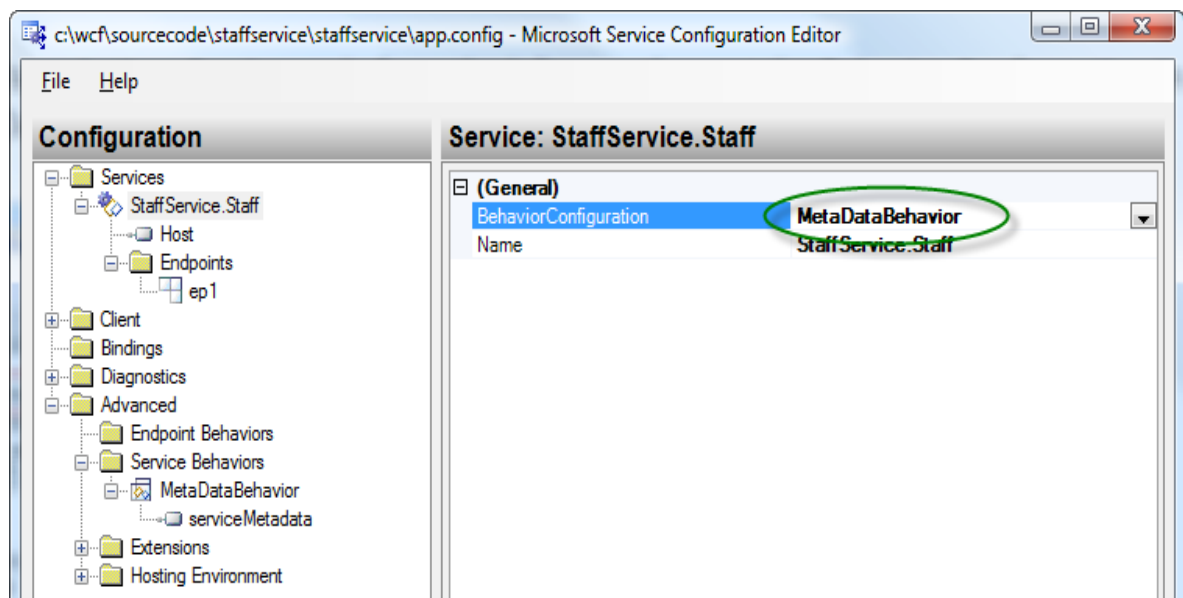


Figure 13 Đặt BehaviorConfiguration cho dịch vụ

5. Cuối cùng là tạo điểm cuối để các ứng dụng có thể kết nối tới dịch vụ để lấy các thông tin về dịch vụ. Bấm chuột phải vào Endpoints vào chọn New Service Endpoint sau đó đặt cấu hình như sau:

- Name: ep2
- Address: mex

- Binding: mexHttpBinding
- Contract: IMetadataExchange

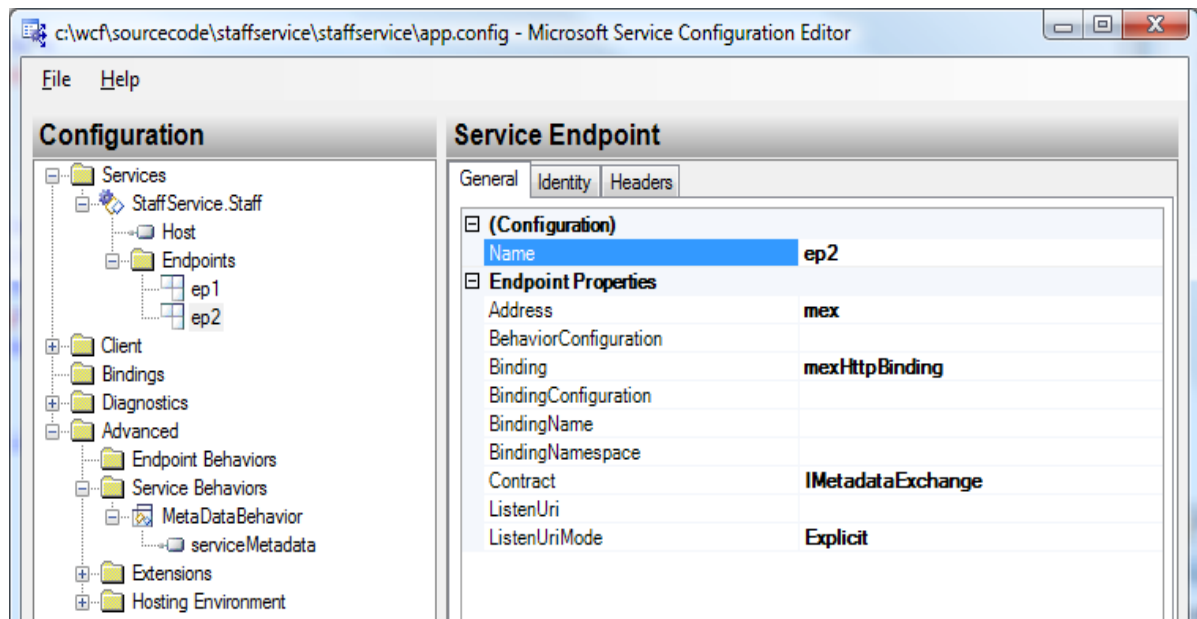


Figure 14 Cấu hình cho điểm cuối

6. Bấm menu Save bạn sẽ có tệp cấu hình như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <system.serviceModel>
    <behaviors>
      <serviceBehaviors>
        <behavior name="MetadataBehavior">
          <serviceMetadata httpGetEnabled="true"
httpGetUrl="http://localhost:8000/StaffService" />
        </behavior>
      </serviceBehaviors>
    </behaviors>
    <services>
      <service behaviorConfiguration="MetadataBehavior"
name="StaffService.Staff">
        <endpoint address="" binding="wsHttpBinding"
bindingConfiguration=""
name="ep1" contract="StaffService.IStaff" />
        <endpoint address="mex" binding="mexHttpBinding"
bindingConfiguration=""
name="ep2" contract="IMetadataExchange" />
      </service>
    </services>
  </system.serviceModel>
</configuration>
```

```

        <host>
            <baseAddresses>
                <add
baseAddress="http://localhost:8000/StaffService" />
            </baseAddresses>
        </host>
    </service>
</services>
</system.serviceModel>
</configuration>

```

7. Giờ bạn có thể chạy dịch vụ. Sẽ không có lỗi gì xảy ra và bạn sẽ có màn hình như sau:

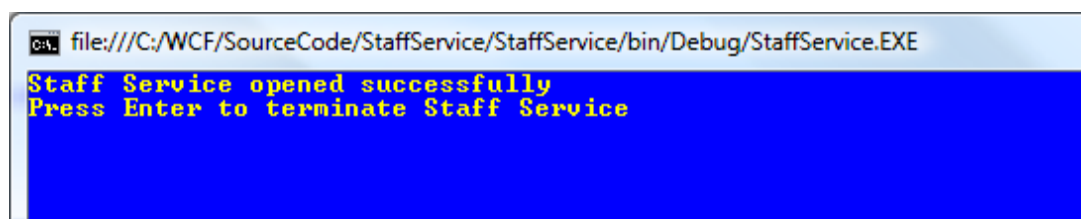


Figure 15 Thực hiện dịch vụ

8. Nếu sử dụng Internet Explorer truy xuất tới địa chỉ <http://localhost:8000/StaffService> bạn sẽ thấy màn hình sau:

Staff Service

You have created a service.

To test this service, you will need to create a client and use it to call the service. You can do this using the svcutil.exe tool from the command line with the following syntax:

```
svcutil.exe http://localhost:8000/StaffService?wsdl
```

This will generate a configuration file and a code file that contains the client class. Add the two files to your client application and use the generated client class to call the Service. For example:

C#

```
class Test
{
    static void Main()
    {
        StaffClient client = new StaffClient();

        // Use the 'client' variable to call operations on the service.

        // Always close the client.
        client.Close();
    }
}
```

Figure 16 Sử dụng IE để xem dịch vụ

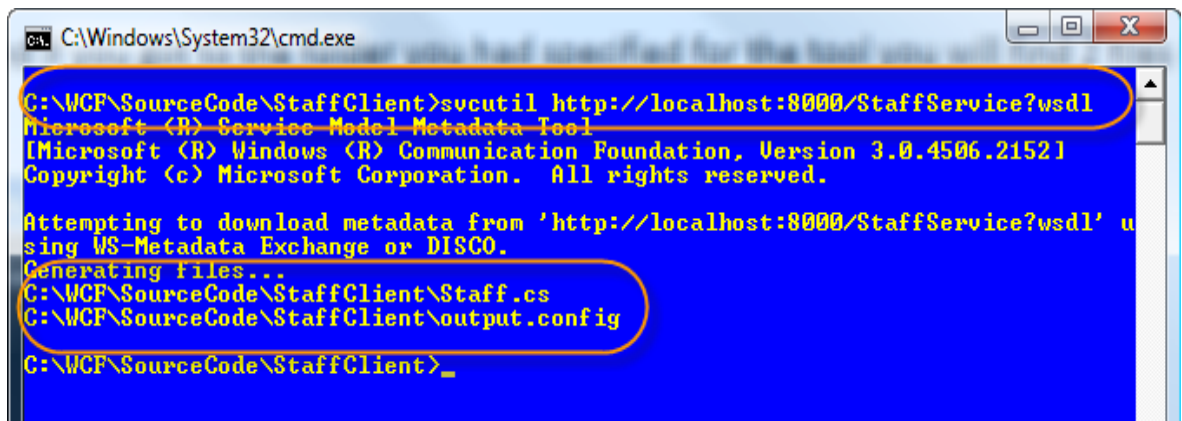
2.2.3 Tạo ứng dụng phía client

Phần này hướng dẫn các bạn tạo ứng dụng phía client để truy xuất các hàm do dịch vụ trên cung cấp. Điểm hay trong WCF là bạn chỉ cần sử dụng một dòng mã nguồn để gọi đến dịch vụ StaffService, chỉ một dòng mà thôi, những thứ khác được tạo ra tự động sử dụng công cụ svcutil.exe.

1. Mở console dòng lệnh và chuyển tới thư mục bạn muốn tạo các tệp, gõ vào lệnh sau và bấm Enter:

```
Svcutil.exe http://localhost:8000/StaffService?wsdl
```

Bạn sẽ thấy kết quả sau:



```
C:\Windows\System32\cmd.exe

C:\WCF\SourceCode\StaffClient>svcutil http://localhost:8000/StaffService?wsdl
Microsoft (R) Service Model Metadata Tool
[Microsoft (R) Windows (R) Communication Foundation, Version 3.0.4506.2152]
Copyright (c) Microsoft Corporation. All rights reserved.

Attempting to download metadata from 'http://localhost:8000/StaffService?wsdl' u
sing WS-Metadata Exchange or DISCO.
Generating files...
C:\WCF\SourceCode\StaffClient\Staff.cs
C:\WCF\SourceCode\StaffClient\output.config
C:\WCF\SourceCode\StaffClient>
```

Figure 17 Kết quả tạo ra khai báo dịch vụ

2. Tạo ra một console project mới đặt tên là StaffClient và thêm tham chiếu tới System.ServiceModel. Thêm vào tệp vừa tạo ra là Staff.cs, đồng thời đổi tên tệp cấu hình thành app.config và thêm vào project. Kết quả như sau:

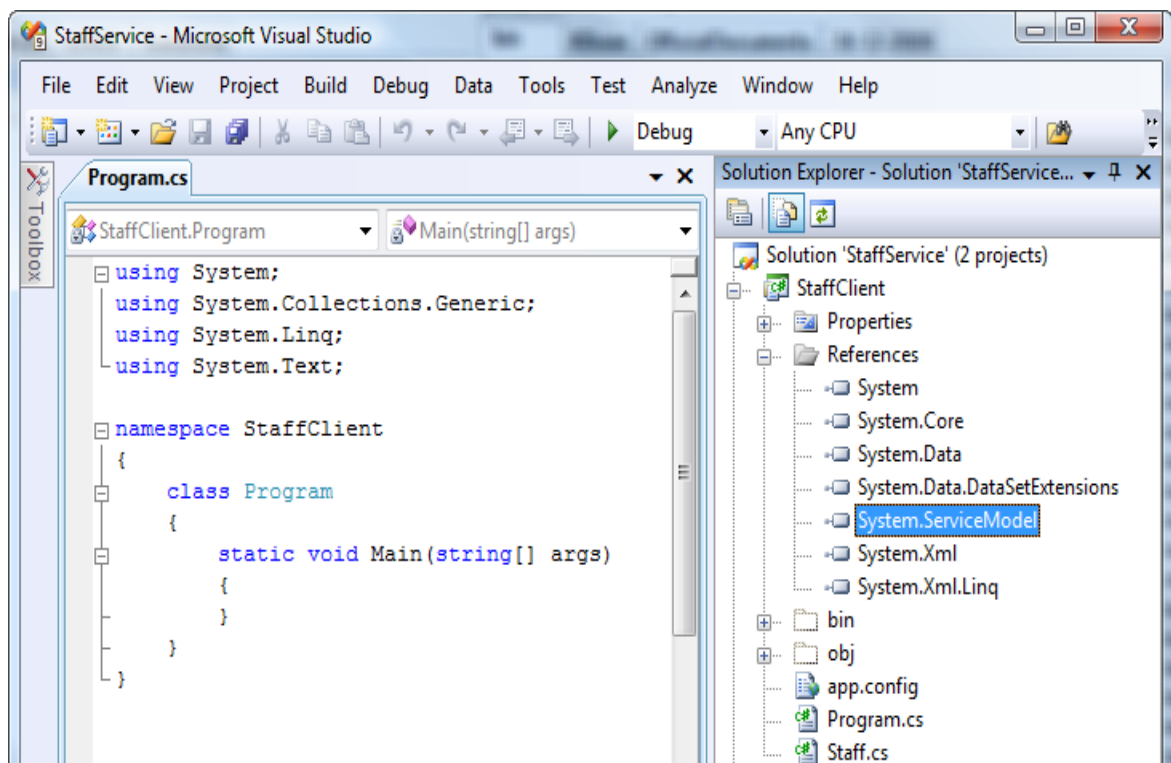
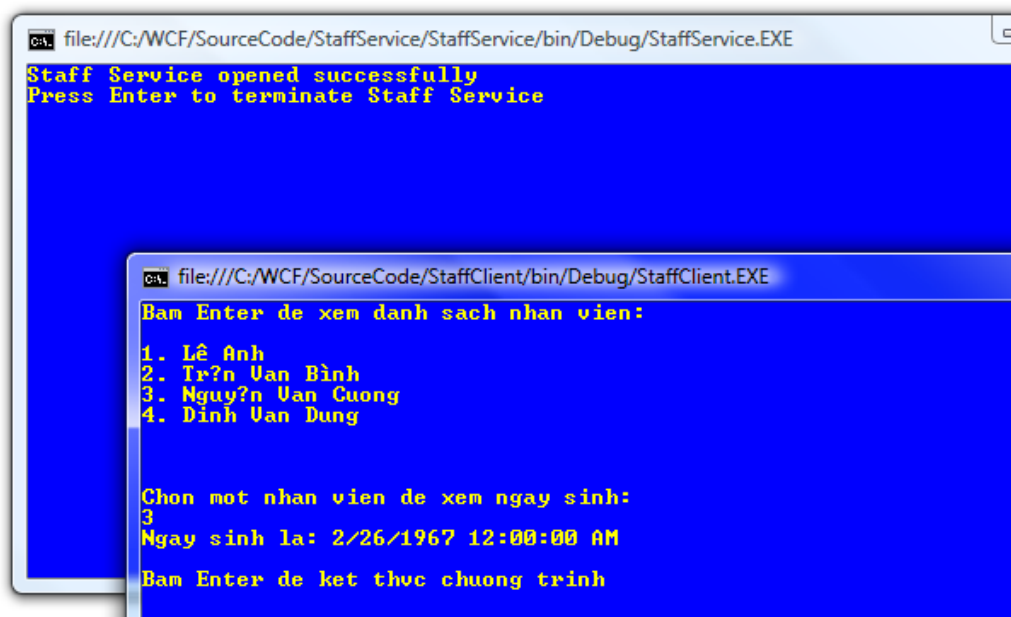


Figure 18 Tạo client

3. Sau khi biên dịch và chạy chương trình bạn được kết quả sau:



```
file:///C:/WCF/SourceCode/StaffService/StaffService/bin/Debug/StaffService.EXE
Staff Service opened successfully
Press Enter to terminate Staff Service

file:///C:/WCF/SourceCode/StaffClient/bin/Debug/StaffClient.EXE
Bam Enter de xem danh sach nhan vien:
1. Lê Anh
2. Tr?n Van Bình
3. Nguy?n Van Cuong
4. Đinh Van Dung

Chon mot nhan vien de xem ngay sinh:
3
Ngay sinh la: 2/26/1967 12:00:00 AM
Bam Enter de ket thuc chuong trinh
```

Figure 19 Kết quả chạy chương trình

2.3 Cách khác để tạo tham chiếu ở client

Ngoài cách thực hiện bằng dòng lệnh như ở trên, ta hoàn toàn có thể sử dụng sự giúp đỡ của công cụ Visual Studio để tạo ra tham chiếu dịch vụ phía client. Cách này thực hiện rất nhanh và rất tiện lợi. Cách làm như sau:

Bước 1. Chọn chuột phải vào project cần thêm tham chiếu, ở đây là StaffClient, sau đó chọn menu là Add Service Reference như hình sau:

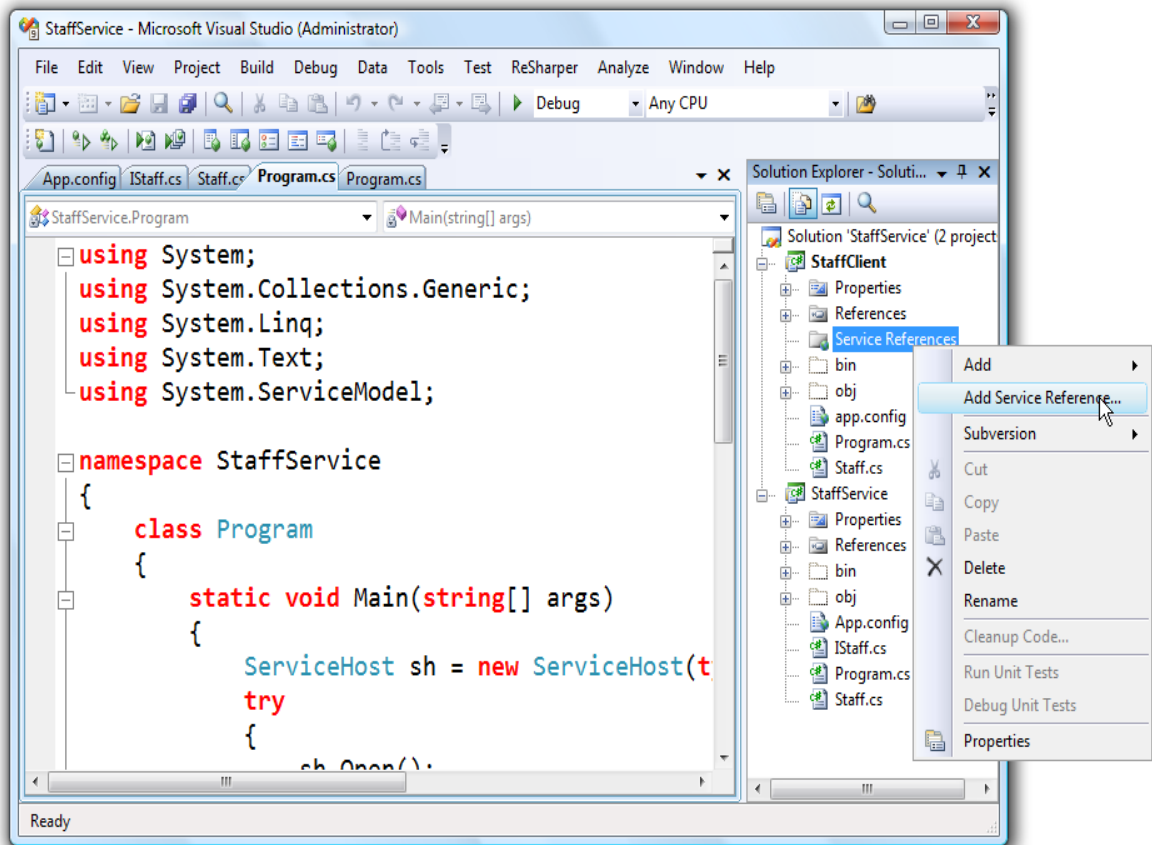


Figure 20 Thêm tham chiếu dịch vụ

Bước 2. Gõ vào địa chỉ của dịch vụ cần thêm, sau đó bấm nút Go, phía dưới đặt tên cho tham chiếu dịch vụ là StaffService như hình dưới.

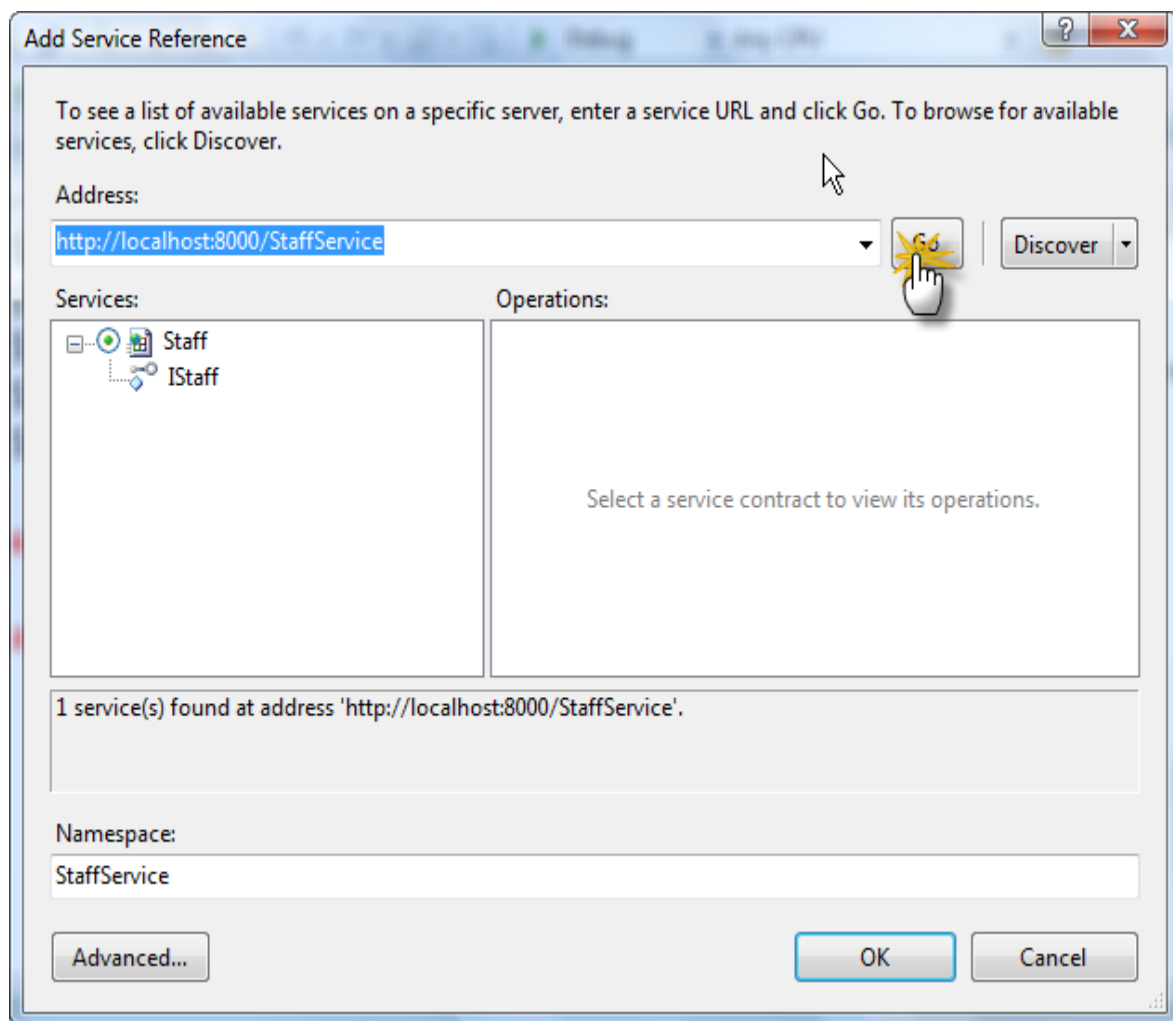


Figure 21 Cấu hình các tham số cho tham chiếu dịch vụ

Sau bước này bạn hoàn toàn có thể sử dụng lớp StaffClient để cài đặt ứng dụng phía client.

3 Câu hỏi ôn tập

1. Liệt kê các mô hình lập trình với WCF

Có 2 mô hình lập trình chủ yếu với WCF là phương pháp hướng đối tượng và phương pháp hướng dịch vụ. Phương pháp hướng đối tượng thông thường được sử dụng khi phát triển ứng dụng ở desktop, còn phương pháp hướng dịch vụ được sử dụng để phát triển các dịch vụ phía server. Mối liên kết giữa hai phương pháp này thông qua các lớp và giao diện (hướng đối tượng) và các thuộc tính WCF trên các lớp hay giao diện đó (hướng dịch vụ)

2. Các phương pháp lập trình với WCF

Có 3 phương pháp lập trình với WCF là phương pháp khai báo, phương pháp hiển hiện, và phương pháp sử dụng tệp tin cấu hình. Thông thường khi làm việc với WCF, ta không sử dụng

riêng biệt một phương pháp nào mà sử dụng kết hợp cả ba phương pháp để đạt được kết quả tốt nhất.

4 Tài liệu tham khảo

1. Programming your first WCF service (URL:
<http://www.myitblog.com/sundararajan/programming-your-first-wcf-service.html>)
2. Your first WCF Service (URL: <http://eng.ahmedelmalt.googlepages.com/wcf02.htm>)