Bài 4

CONTRACTS TRONG WINDOWS COMMUNICATION FOUNDATION

Mục lục

1 Contracts trong Windows Communication Foundation 2
1.1 Service Contracts 3
1.1.1 Thuộc tính ServiceContract 5
1.1.2 Thuộc tính OperationContract 6
1.2 Data Contracts 8
1.2.1 Thuộc tính DataContract 8
1.3 Message Contracts 9
1.3.1 Thuộc tính MessageContract 9
1.3.2 Thuộc tính MessageHeader 10
1.3.3 Thuộc tính MessageBodyMember 10
2 Lập trình với các contracts trong Windows Communication Foundation 10
3 Câu hỏi ôn tập 17
4 Tài liệu tham khảo 18

Trong các bài trước, chúng ta đã nói đến các địa chỉ (address) và bindings trong Windows Communication Foundation. Bài này sẽ giới thiệu tới các bạn về thành phần thứ ba rất quan trọng của WCF, đó là các contract (giao kèo). Nội dung của bài này sẽ tập trung vào thảo luận các loại contract, một số ví dụ để định nghĩa contract và lập trình với contract.

Tất cả các ví dụ mà bạn đã thấy và làm việc ở các bài trước cho đến giờ đều sử dụng và cài đặt các contract. Mặc dù đã làm việc với contract, nhưng chúng ta chưa nói một cách chi tiết về contract là gì, có các loại contract nào và làm thế nào để làm việc với chúng. Trong bài này chúng ta sẽ thảo luận các vấn đề trên theo thứ tự sau:

- Các loại (kiểu) của contract trong WCF
- Làm thế nào để định nghĩa các kiểu contract khác nhau
- Ví dụ về các contract

1 Contracts trong Windows Communication Foundation

Các contract trong Windows Communication Foundation cung cấp khả năng làm việc đa môi trường khi liên lạc với các máy khách. Các máy khách và dịch vụ thoả thuận với nhau về các contract như kiểu của các hành động và các cấu trúc dữ liệu mà các bên sẽ sử dụng trong quá trình liên lạc qua lại với nhau. Nếu không có contract thì không thể thực hiện liên lạc được do không có sự thoả thuận thống nhất giữa các bên.

Khi định nghĩa một dịch vụ trong Windows Communication Foundation ta sử dụng ba kiểu contract cơ bản sau:

- **Service Contract (Contract dịch vụ)**: Định nghĩa các phương thức của một dịch vụ, thực chất là các hành động mà client có thể sử dụng ở các điểm cuối (endpoint)
- **Data Contract (Contract dữ liệu)**: Định nghĩa các kiểu dữ liệu được sử dụng ở các phương thức của dịch vụ
- **Message Contract (Contract bản tin)**: Cung cấp khả năng để điều khiển các đầu đề bản tin trong quá trình tạo ra các bản tin

Cũng giống như các thành phần khác trong WCF, các contract được định nghĩa sử dụng các khái niệm mà bạn đã quen thuộc. Các contract được định nghĩa sử dụng các lớp và giao diện, bằng cách thêm vào các thuộc tính mô tả (attribute) cho các lớp và giao diện đó.

Có thể hình dung mối liên hệ giữa các contract với .NET Framework CLR một cách gần đúng như sau:

- **Service Contracts**: Cung cấp thông tin để thực hiện ánh xạ từ CLR sang WSDL (Ngôn ngữ mô tả dịch vụ Web – Web Service Description Language)

- **Data Contracts**: Cung cấp thông tin để thực hiện ánh xạ từ CLR sang XSD (Định nghĩa schema XML – XML Schema Definition)
- **Message Contracts**: Dùng để miêu tả cấu trúc của bản tin SOAP (Giao thức truy nhập đối tượng đơn giản – Simple Object Access Protocol)

1.1 Service Contracts

Như đã nói ở phần trên, một contract dịch vụ định nghĩa các hành động hoặc phương thức có ở điểm cuối dịch vụ và được đưa ra để máy khách có thể sử dụng. Nó còn định nghĩa một cách cơ bản các phép trao đổi bản tin như việc bản tin sẽ thế nào trong yêu cầu/trả lời hay trong liên lạc đơn công hoặc song công.

Contract dịch vụ đưa ra một số thông tin cho các máy khách đủ để cho máy khách có thể biết được dịch vụ này có thể cung cấp những gì. Những thông tin đó bao gồm:

- Các kiểu dữ liệu trong bản tin
- Vị trí của các phương thức – hành động
- Thông tin về giao thức, định đạng lưu dữ liệu để đảm bảo cho liên lạc thành công
- Nhóm các hành động
- Mẫu trao đổi bản tin (Message exchange pattern – MEP)

Như trên đã nói, để định nghĩa contract dịch vụ, ta sử dụng các thuộc tính mô tả cho một lớp hay giao diện. Thuộc tính mô tả một contract dịch vụ là `ServiceContract`. Ví dụ sau định nghĩa một giao diện như là một contract dịch vụ:

```
[ServiceContract]
public interface IStaffInformation
{
}
```

Sau đó để định nghĩa các hành động (phương thức) cho dịch vụ ta sử dụng phần mô tả là `OperationContract` cho các phương thức của giao diện như ví dụ dưới đây:

```
        [OperationContract]
        bool HasPerson(int staffId);

        [OperationContract]
        string GetPersonName(int staffId);
```

Gộp các phần lại ta sẽ được một định nghĩa hoàn chỉnh cho một contract dịch vụ:

```
    [ServiceContract]
    public interface IStaffInformation
```

```
{
    [OperationContract]
    bool HasPerson(int staffId);

    [OperationContract]
    string GetPersonName(int staffId);
}
```

Sau khi có được định nghĩa cho dịch vụ, ta có thể cài đặt một lớp cho giao diện trên, trong đó định nghĩa các hàm (phương thức) thực hiện theo logic của dịch vụ.

```
public class StaffInformation : IStaffInformation
{
    public bool HasPerson(int staffId)
    {
        // Làm việc gì đó ở đây, và trả về kết quả
    }

    public string GetPersonName(int staffId)
    {
        // Làm việc gì đó ở đây, và trả về kết quả
    }
}
```

Trong trường hợp bạn không muốn định nghĩa dịch vụ ở phần giao diện, bạn hoàn toàn có thể định nghĩa dịch vụ ở trong lớp cài đặt dịch vụ đó như sau:

```
[ServiceContract]
public class StaffInformation : IStaffInformation
{
    [OperationContract]
    public bool HasPerson(int staffId)
    {
        // Làm việc gì đó ở đây, và trả về kết quả
    }

    [OperationContract]
    public string GetPersonName(int staffId)
    {
        // Làm việc gì đó ở đây, và trả về kết quả
    }
}
```

Trong khi định nghĩa contract dịch vụ, chúng ta đã sử dụng hai lớp thuộc tính để mô tả là `ServiceContract` và `OperationContract`. Hai thuộc tính này có rất nhiều tham số kèm theo, tuy nhiên trong nhiều trường hợp, chúng ta hoàn toàn có thể sử dụng các giá trị mặc định của hai thuộc tính này để định nghĩa contract dịch vụ. Trong một số trường hợp khi chúng ta muốn điều khiển sâu hơn về các thông tin liên quan đến dịch vụ, ta có thể đưa thêm các tham số cho hai thuộc tính này.

1.1.1 Thuộc tính ServiceContract

Thuộc tính `ServiceContract` được áp dụng cho việc mô tả các giao diện hoặc các lớp để định nghĩa một contract dịch vụ. Thuộc tính này có các tham số sau:

Tên tham số	Mô tả

CallbackContract	Thiết lập/Trả về kiểu của callback contract khi liên lạc ở chế độ song công. Khi liên lạc giữa máy khách và dịch vụ được thiết lập, tham số này chỉ ra rằng máy khách cần phải đợi lời gọi hàm từ phía dịch vụ thông qua kiểu của callback contract đã định nghĩa. Ví dụ sau mô tả cách sử dụng tham số `CallbackContract`
	`[ServiceContract(CallbackContract=typeof(IClientContract))]`
	`public class StaffInformation : IStaffInformation`
	`{`
	`....`
	`}`
	Trong ví dụ trên, dịch vụ quy định callback contract phải có kiểu là `IClientContract`

ConfigurationName	Thiết lập/Trả về tên được sử dụng để xác định thành phần dịch vụ trong tệp tin cấu hình. Ví dụ về việc sử dụng tham số `ConfigurationName` như sau:
	`[ServiceContract(ConfigurationName="DichVu")]`
	`public class StaffInformation : IStaffInformation`
	`{`
	`....`
	`}`
	Và ở tệp tin cấu hình ta có phần định nghĩa thành phần dịch vụ với tên là `DichVu`
	`<?xml version="1.0" encoding="utf-8" ?>`
	`<configuration>`

	<system.serviceModel> <services> <service name="DichVu"> </service> </services> </system.serviceModel> </configuration>
Name	Thiết lập/Trả về tên của thành phần <portType> trong WSDL. Giá trị mặc định cho tham số này chính là tên của giao diện hay lớp có gắn thuộc tính ServiceContract. Tham số này được sử dụng trong trường hợp ta muốn thay đổi tên của thành phần <portType> hoặc muốn giữ nguyên tên của thành phần <portType> nhưng lại đổi tên của giao diện hay lớp định nghĩa dịch vụ.
Namespace	Thiết lập/Trả về namespace (không gian tên) cho thành phần <portType> trong WSDL. Giá trị mặc định cho tham số này là http://tempuri.org
ProtectionLevel	Quy định yêu cầu về mức bảo vệ trong binding. Việc quy định này bao gồm quy định về mã hoá, chữ ký điện tử tại các điểm cuối của dịch vụ

SessionMode	Quy định kiểu hỗ trợ cho các phiên làm việc tin cậy mà một dịch vụ đòi hỏi hoặc hỗ trợ. Ví dụ có thể định nghĩa đòi hỏi phải hỗ trợ phiên làm việc tin cậy cho dịch vụ như sau:
	[ServiceContract(SessionMode=SessionMode.Required)]
	public class StaffInformation : IStaffInformation
	{
	....
	}

1.1.2 Thuộc tính OperationContract

Thuộc tính OperationContract được gắn với các phương thức trong các giao diện hay các lớp. Chỉ các phương thức được gắn thuộc tính OperationContract mới được coi là phương thức của dịch vụ. Các tham số có thể sử dụng cho thuộc tính này như sau:

Tên tham số	Mô tả
Action	Quy định hành động để chỉ ra một cách duy nhất phương thức này. WCF phân phối các bản tin yêu cầu với các phương thức dựa trên các hành

	động của chúng.
AsyncPattern	Chỉ ra rằng phương thức được cài đặt hoặc có thể gọi theo cách dị bộ sử dụng cặp phương thức bắt đầu bởi Begin và End
IsInitiating	Quy định phương thức này có phải là phương thức để khởi tạo trong một phiên hay không
IsOneWay	Chỉ ra rằng phương thức này chỉ chứa một bản tin đầu vào duy nhất. Phương thức không có bản tin trả về.
IsTerminating	Quy định xem liệu WCF có kết thúc phiên làm việc hiện tại sau khi phương thức này thực hiện xong hay không
Name	Quy định tên cuối cùng của phương thức sẽ có trong dịch vụ. Giá trị mặc định của tham số này là tên của phương thức
ProtectionLevel	Quy định sự bảo vệ ở mức bản tin mà một phương thức yêu cầu khi thực hiện
ReplyAction	Quy định hành động của bản tin trả lời cho phương thức này

Ví dụ sau biểu diễn việc sử dụng các tham số của OperationContract để quy định thứ tự thực hiện của các phương thức khi làm việc với dịch vụ. Theo ví dụ này, quy trình để máy khách liên lạc với dịch vụ như sau.

1. Đầu tiên máy khách cần phải gọi hàm `Login` để đăng nhập
2. Sau đó gọi hàm `HasPerson` hoặc `GetPersonName` để lấy thông tin về nhân viên
3. Cuối cùng gọi hàm `Logout` để đăng xuất

```
[ServiceContract]
public interface IStaffInformation
{
    [OperationContract(IsInitiating=true, IsTerminating=false)]
    void Login(string userName, string password);
    [OperationContract]
    bool HasPerson(int staffId);
    [OperationContract]
    string GetPersonName(int staffId);
    [OperationContract(IsInitiating=false, IsTerminating=true)]
    void Logout();
```

```
}
```

Nếu máy khách không thực hiện theo thứ tự trên thì dịch vụ sẽ báo lỗi và không thực hiện.

1.2 Data Contracts

Một cách đơn giản thì một contract dữ liệu mô tả dữ liệu cần trao đổi. Trước khi máy khách và dịch vụ thực hiện liên lạc thì chúng phải đồng ý với nhau về kiểu dữ liệu trao đổi, đó là contract dữ liệu. Điều quan trọng là khi thực hiện liên lạc, máy khách và dịch vụ hoàn toàn không cần phải chung nhau cùng các kiểu dữ liệu mà chúng chỉ cần chung nhau các contract dữ liệu. Contract dữ liệu định nghĩa cách serialized và deserialized cho từng tham số và kiểu trả về.

Quá trình serialization dữ liệu là quá trình chuyển một cấu trúc dữ liệu thành một định dạng có thể dùng trong liên lạc hoặc gửi qua đường truyền. Ví dụ đọc dữ liệu từ cơ sở dữ liệu sau đó chuyển nó thành một chuỗi các byte và gửi qua đường truyền.

Quá trình deserialization là quá trình ngược lại với quá trình serialization. Quá trình này nhận dữ liệu từ đường truyền và chuyển ngược thành cấu trúc dữ liệu.

Để định nghĩa contract dữ liệu, ta sử dụng thuộc tính `DataContract` và `DataMember`.

1.2.1 Thuộc tính DataContract

Cũng giống như thuộc tính ServiceContract, thuộc tính DataContract được gắn vào các lớp để định nghĩa contract dữ liệu. Ví dụ sau định nghĩa một contract dữ liệu

```
[DataContract]
public class Person
{
...
}
```

Sau khi định nghĩa contract dữ liệu, bạn có thể định nghĩa các thành viên của một contract dữ liệu. Những thành viên này thực chất là các trường hay thuộc tính trong lớp của bạn. Ví dụ sau đây định nghĩa một số thành viên trong contract dữ liệu.

```
[DataContract]
public class Person
{
    [DataMember]
    int StaffId;
    [DataMember]
    string FullName;
    [DataMember]
```

```
        int Age;
    }
```

Sau khi định nghĩa xong contract dữ liệu và các thành viên có trong contract dữ liệu, ta có thể sử dụng contract dữ liệu này trong khai báo phương thức của contract dịch vụ như sau:

```
    [ServiceContract]
    public interface IStaffInformation
    {
        [OperationContract]
        public void AddPerson(Person person);
    }
```

1.3 Message Contracts

Contract bản tin cho phép bạn điều khiển toàn bộ định dạng của các bản tin SOAP. Trong phần lớn các trường hợp, bạn không cần sử dụng mức điều khiển này, do các contract dữ liệu đã cho phép bạn điều khiển theo yêu cầu. Tuy vậy, vẫn có một vài trường hợp bạn cần đến mức điều khiển bằng các contract bản tin.

Có lẽ lý do lớn nhất khiến bạn cần tới contract bản tin là tính làm việc liên môi trường. Một contract bản tin cung cấp các mức độ làm việc liên môi trường bạn cần khi liên lạc với các máy khách hoặc các hệ thống khác sử dụng một schema hay WSDL nào đó.

Contract bản tin cho phép bạn có nhiều lựa chọn hơn trong việc định dạng các tham số trong các bản tin SOAP ví như liệu thông tin có hay không có ở trong nội dung bản tin hay đầu đề bản tin. Để định nghĩa contract bản tin, ta sử dụng các lớp thuộc tính mô tả sau: `MessageContract`, `MessageHeader`, và `MessageBodyMember` cùng với các tham số tương ứng với từng lớp.

1.3.1 Thuộc tính MessageContract

Các contract bản tin được định nghĩa bằng cách gắn thuộc tính `MessageContract` vào các lớp. Sau đó cần phải xác định rõ cho từng bản tin các thuộc tính `MessageHeader` và `MessageBodyMember`. Ví dụ sau biểu diễn một contract bản tin đơn giản.

```
    [MessageContract]
    public class Person
    {
        [MessageHeader]
        int StaffId;
        [MessageBodyMember]
        string FullName;
        [MessageBodyMember]
```

```
        int Age;
    }
```

1.3.2 Thuộc tính MessageHeader

Thuộc tính `MessageHeader` được gắn với các trường hoặc thuộc tính của một kiểu. Những trường hay thuộc tính này sau đó sẽ được ánh xạ thành các đầu đề bản tin SOAP.

1.3.3 Thuộc tính MessageBodyMember

Thuộc tính `MessageBodyMember` được gắn với các trường hoặc thuộc tính của một kiểu. Những trường hay thuộc tính này sau đó sẽ được ánh xạ thành phần nội dung của bản tin SOAP.

2 Lập trình với các contracts trong Windows Communication Foundation

Trong phần này chúng ta sẽ tạo ra một dịch vụ đơn giản để quản trị thông tin về nhân viên trong công ty. Dịch vụ cho phép chúng ta:

1. Thêm mới một nhân viên vào cơ sở dữ liệu
2. Xoá một nhân viên khỏi cơ sở dữ liệu
3. Sửa thông tin về nhân viên trong cơ sở dữ liệu
4. Lấy thông tin về nhân viên
5. Lấy danh sách tất cả các nhân viên

Bước 1. Mở Visual Studio 2008, tạo một project mới với kiểu là WCF Service Library, đặt tên là ContractSample

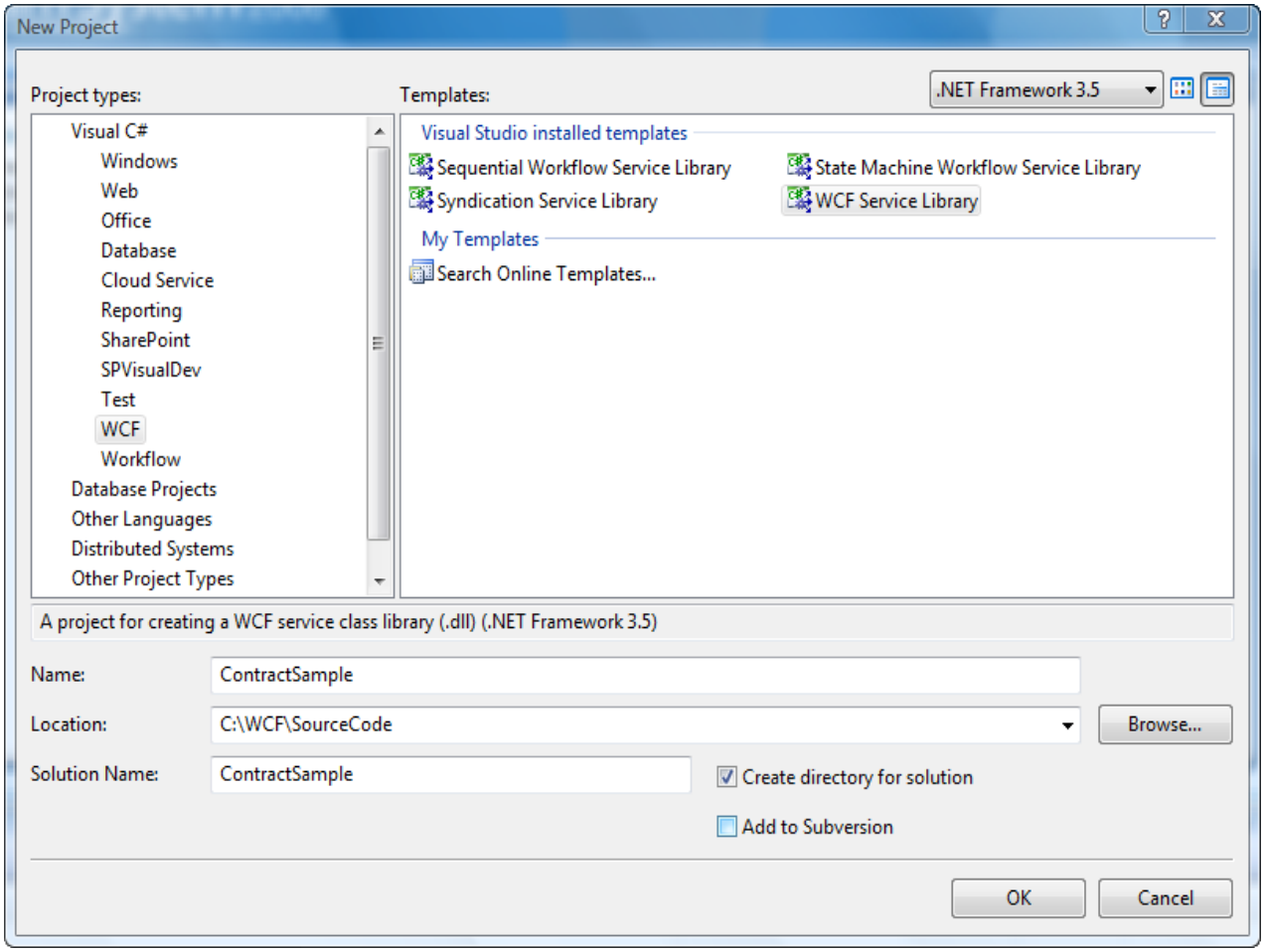

Figure 1 Tạo WCF service library

Bước 2. Xoá tệp tin IService1.cs và Service1.cs trong solution, sau đó chọn chuột phải vào project ContractSample, chọn Add -> New Item và chọn WCF Service, đặt tên cho service này là StaffInfomation, xem hình dưới.

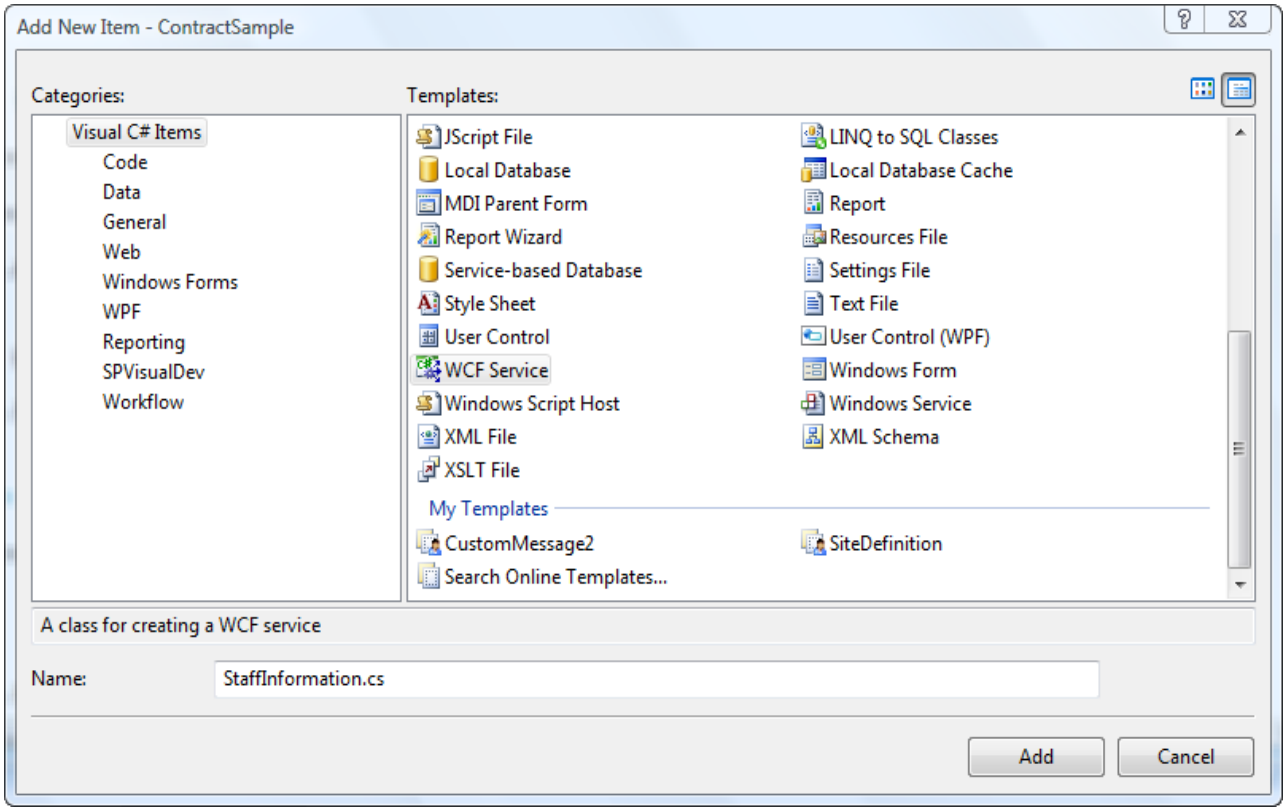

Figure 2 Thêm mới dịch vụ StaffInformation

Bước 3. Mở tệp IStaffInformation, thêm vào đoạn mã nguồn sau:

```
using System.Runtime.Serialization;
using System.ServiceModel;

namespace ContractSample
{
    // định nghĩa contract dữ liệu
    [DataContract]
    public class Person
    {
        [DataMember]
        public int personId;
        [DataMember]
        public string FullName;
        [DataMember]
        public int Age;
    }

    // định nghĩa contract dịch vụ
```

```
    [ServiceContract]
    public interface IStaffInformation
    {
        [OperationContract]
        bool HasPerson(int personId);
        [OperationContract]
        Person GetPerson(int personId);
        [OperationContract]
        void AddPerson(Person person);
        [OperationContract]
        void EditPerson(int personId, Person person);
        [OperationContract]
        void DeletePerson(int personId);
        [OperationContract]
        Person[] GetAll();
    }
}
```

Bước 4. Mở tệp StaffInformation.cs, thay toàn bộ mã nguồn trong tệp đó bằng đoạn mã nguồn sau:

```
using System.Collections.Generic;
using System.Linq;
using System.ServiceModel;

namespace ContractSample
{
    public class StaffInformation : IStaffInformation
    {
        List<Person> personCollection = new List<Person>();

        public bool HasPerson(int personId)
        {
            return personCollection.Any(x => x.personId == personId);
        }

        public Person GetPerson(int personId)
        {
            return personCollection.FirstOrDefault(x => x.personId ==
personId);
        }
```

```
        public void AddPerson(Person person)
        {
            personCollection.Add(person);
        }

        public Person[] GetAll()
        {
            return personCollection.ToArray();
        }

        public void EditPerson(int personId, Person person)
        {
            Person p = GetPerson(personId);
            if (p != null)
            {
                p.FullName = person.FullName;
                p.Age = person.Age;
            }
        }

        public void DeletePerson(int personId)
        {
            Person p = GetPerson(personId);
            if (p != null)
            {
                personCollection.Remove(p);
            }
        }
    }
}
```

Bước 5. Sửa tệp tin cấu hình App.config, thay bằng cấu hình sau:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <system.web>
    <compilation debug="true" />
  </system.web>
  <system.serviceModel>
    <services>
      <service
behaviorConfiguration="ContractSample.StaffInformationBehavior"
```

```
        name="ContractSample.StaffInformation">
        <endpoint address="" binding="wsHttpBinding"
contract="ContractSample.IStaffInformation">
          <identity>
            <dns value="localhost" />
          </identity>
        </endpoint>
        <endpoint address="mex" binding="mexHttpBinding"
contract="IMetadataExchange" />
        <host>
          <baseAddresses>
            <add
baseAddress="http://localhost:8731/ContractSample/StaffInformation/" />
          </baseAddresses>
        </host>
      </service>
    </services>
    <behaviors>
      <serviceBehaviors>
        <behavior name="ContractSample.StaffInformationBehavior">
          <serviceMetadata httpGetEnabled="true" />
          <serviceDebug includeExceptionDetailInFaults="false" />
        </behavior>
      </serviceBehaviors>
    </behaviors>
  </system.serviceModel>
</configuration>
```

Vậy là bạn đã có một dịch vụ quản trị thông tin về các nhân viên trong công ty. Bạn có thể sử dụng chương trình WCF Test Client để thực hiện quản trị thông tin về nhân viên. Chi tiết về cách sử dụng WCF Test Client, xem bài mở đầu. Ví dụ một phiên quản trị như sau:

1. Thêm mới nhân viên

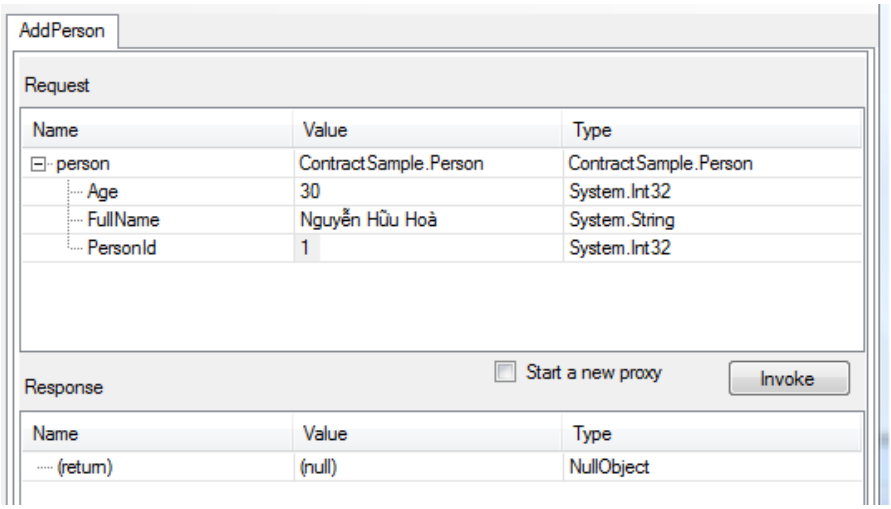

Figure 3 Thêm mới một nhân viên

2. Lấy danh sách tất cả nhân viên

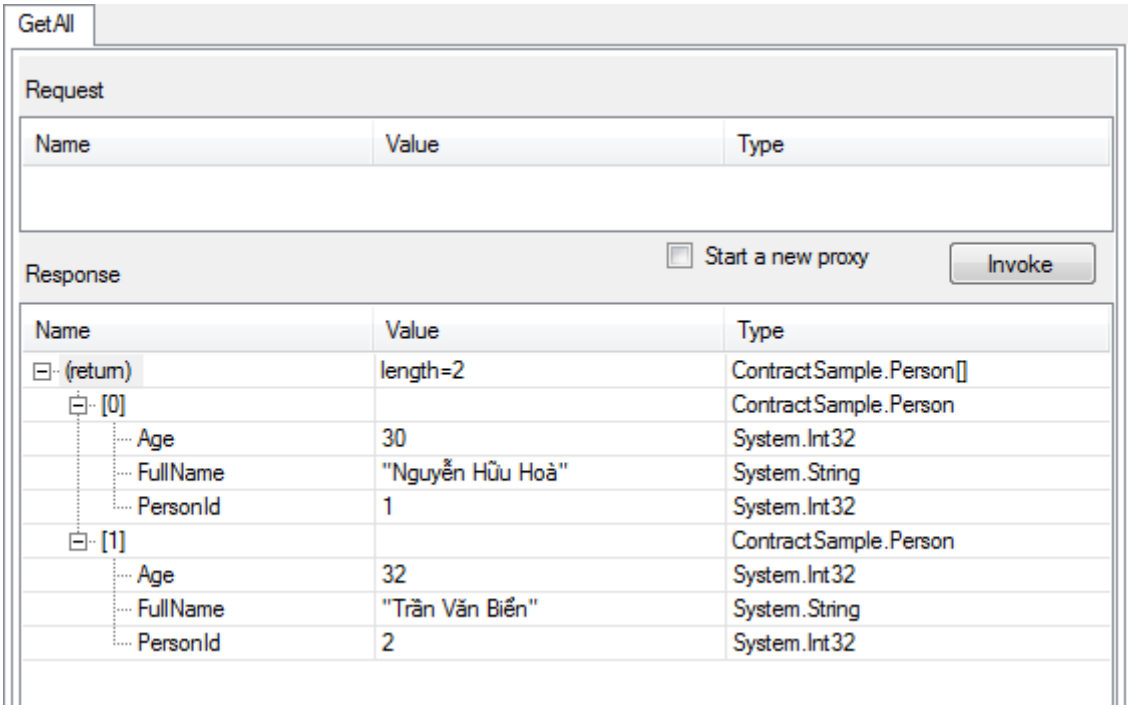

Figure 4 Lấy danh sách các nhân viên

3. Sửa thông tin một nhân viên

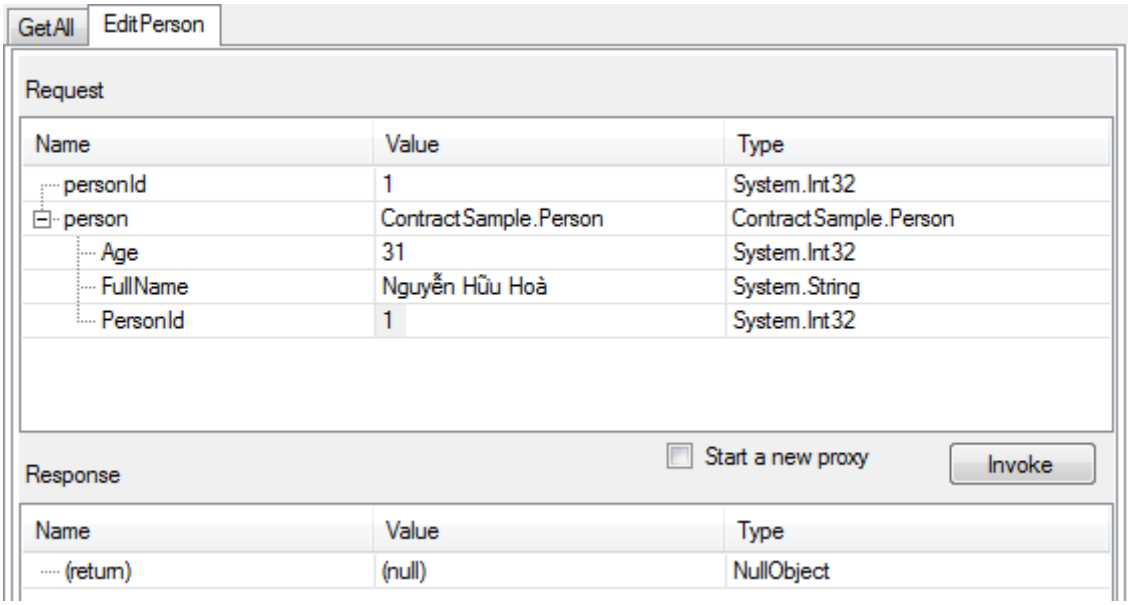

Figure 5 Sửa thông tin về nhân viên

4. Lấy thông tin về một nhân viên

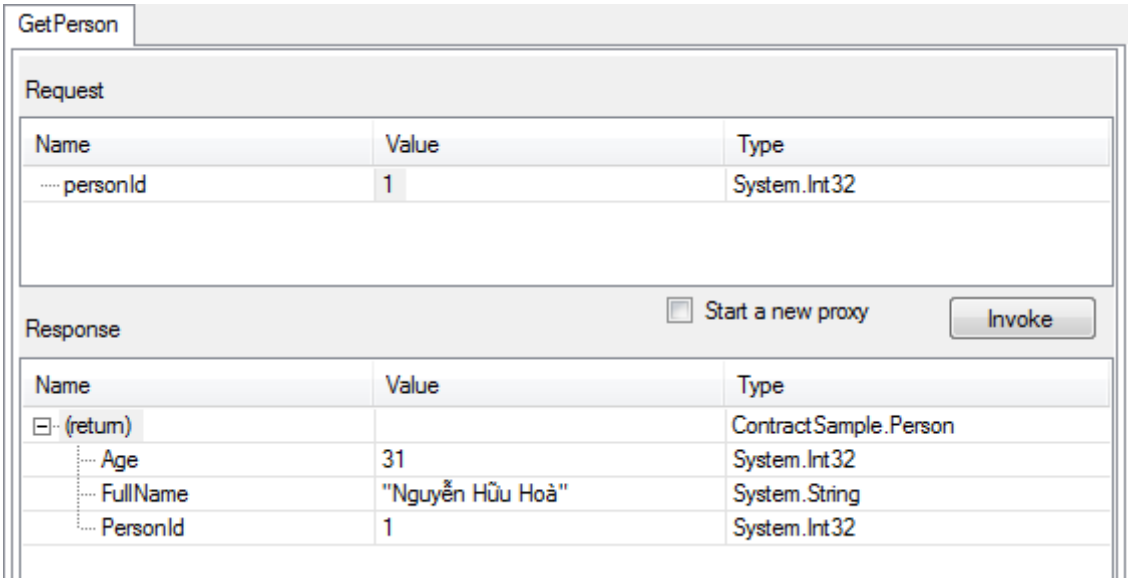

Figure 6 Lấy thông tin về một nhân viên

3 Câu hỏi ôn tập

1. Liệt kê các kiểu contract có trong WCF

Có các kiểu contract sau:

- Service Contract
- Data Contract

- Message Contract

2. Để định nghĩa một service contract, cần phải làm những gì?

Để định nghĩa một service contract ta cần làm các bước sau:

- Tạo một lớp (class) hoặc một giao diện (interface) để khai báo dịch vụ
- Gắn cho lớp hoặc giao diện đó thuộc tính `ServiceContract`
- Thêm vào thuộc tính `OperationContract` cho các phương thức cần có của dịch vụ

4 Tài liệu tham khảo

1. Introduction to Building Windows Communication Foundation Services (URL: http://msdn.microsoft.com/en-us/library/aa480190.aspx)
2. WCF Essentials—Service Contract (URL: http://en.csharp-online.net/WCF_Essentials-Service_Contract)
3. ServiceContractAttribute class (URL: http://msdn.microsoft.com/en-us/library/system.servicemodel.servicecontractattribute.aspx)
4. DataContractAttribute class (URL: http://msdn.microsoft.com/en-us/library/system.runtime.serialization.datacontractattribute.aspx)
5. MessageContractAttribute class (URL: http://msdn.microsoft.com/en-us/library/system.servicemodel.messagecontractattribute.aspx)