Working with Structured Data in Microsoft Office SharePoint Server 2007 (Part2): Exposing LOB Data

Applies to: Microsoft Office SharePoint Server 2007. SQL Server 2005 /2008

Explore different options you have to work with structured data in a high volume while you need to perform complex queries and actions against such data ranging from authoring, approval and landing information on Web Part pages, all the way down to the physical storage. This blog post is part 2 of a blog post series that I am planning to write on this topic. (10 printed pages)

Reza Alirezaei , Microsoft Office Server 2007 MVP
3/23/2009

* For comments please see http://blogs.devhorizon.com/reza/?p=842

Content:

Introduction: SharePoint is a composite application platform meaning that it can be layered on the top of a variety of Microsoft or non-Microsoft data repositories and interact with them either in real time or in background. An issue that arises in the early stages of the design of your SharePoint applications is that how best to make this interaction happen.

Well, Web Services are part of the .NET framework and therefore they are available to SharePoint as well. They provide a cross-platform solution for exchanging data between SharePoint and other system s in a distributed model. As you will see later in this blog post, I will create two service interfaces to expose LOB data to SharePoint and its complementary citizens such as InfoPath forms, Data View Web part and BDC.

Important
Throughout the upcoming blog posts, I will be demonstrating different functionalities that rely on the endpoints we will review in this blog post. This blog post is not meant to walk you through SharePoint OOTB Web services or how to write custom Web services to call into the SharePoint object model. This will be covered in great details in my upcoming posts.

Coding the Northwind Web Service:

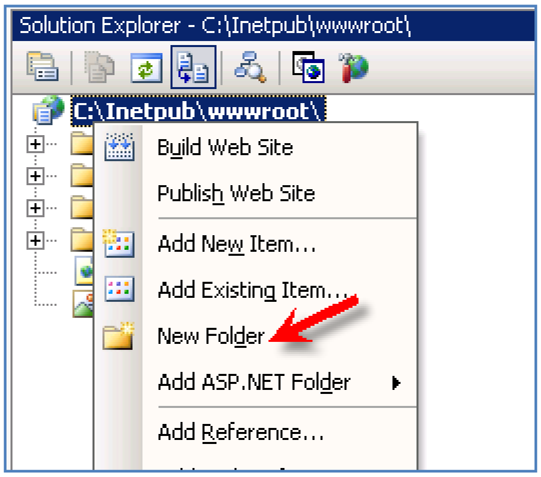
For the sake of brevity and for demonstration purpose, we will create an asmx Web service to expose Northwind data to SharePoint. If you are a WCF guru or you would hate asmx Web services for all their limitations, I am sure you know how to convert them to WCF or WSE services blindfolded!

All right, Let's just get busy and create the Northwind Web service first.

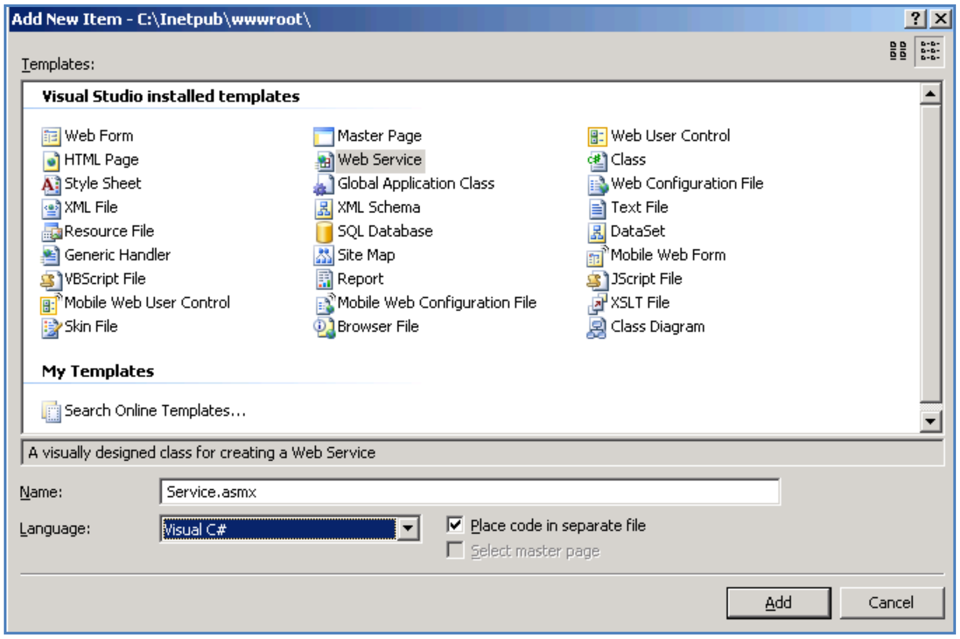
1. Launch Visual Studio
2. From the File menu, select Open and choose Web Site
3. Select File System and then navigate to: *C:\Inetpub\wwwroot\*

Important
You can create this web service in *C:\Program Files\Common Files\Microsoft Shared\web server extensions\12\TEMPLATE\LAYOUTS or C:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\12\ISAPI* folder. Web services in ISAPI folder can be virtualized, context aware and easily discovered no matter how deep you call them in the overall site hierarchy. Depending on the implemented logic in your Web services, the context may or may not be an important thing. In Northwind Web service SharePoint context is not important , because SharePoint is simply the consumer of this service. When writing a Web service that calls into te SharePoint object model , all you need to do is to host it on the WFE server(es) where SharePoint object model is available. If this is how you like to develop your Web services, it's <u>super important</u> to pass in the right URL to Web methods and construct the right context at run time, otherwise you will end up in other contexts.

4. Click Open
5. In the Solution Explorer, right-click on the web site and choose New Folder



6. Rename this folder to: *NorthwindService*
7. Right-click on *NorthwindService* and choose Add New Item
8. From the Visual Studio installed templates list choose Web Service
9. In the Name box, rename this to: *Service.asmx*



A visually designed class for creating a Web Service

Name: Service.asmx

Language: Visual C#

[✓] Place code in separate file

[ ] Select master page

10. Uncheck the option “Place code in a separate file” and click Add
11. Add the following web method to retrieve the top 10 products from the Products table in the Northwind database.

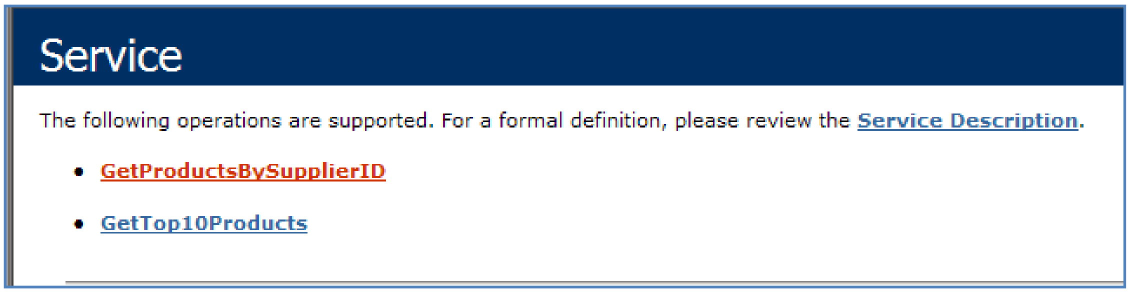
```
[WebMethod]
public DataSet GetTop10Products()
  {
    SqlConnection cn = new SqlConnection("Data

         Source=LITWAREDEMO; Integrated Security=SSPI;Initial
   Catalog=Northwind");
         string strProductsOrigSQL = "SELECT top 10 productname FROM
         Products";
         SqlDataAdapter daProducts = new SqlDataAdapter(strProductsOrigSQL,
                cn);
         DataSet Ds = new DataSet();
         cn.Open();
         daProducts.Fill(Ds, "Products");
         cn.Close();
         cn = null;
         daProducts = null;
         return Ds;
  }
```

12. Add the following web method to retrieve the associated products for the selected supplier.

```
[WebMethod]
public DataSet GetProductsBySupplierID(int suppID)
 {
   SqlConnection cn = new SqlConnection("Data Source=LITWAREDEMO;
Integrated Security=SSPI;Initial Catalog=Northwind");
   string strProductsOrigSQL = "SELECT * FROM Products Where SupplierID
     =" + suppID;
        SqlDataAdapter daProducts = new SqlDataAdapter(strProductsOrigSQL,
           cn);
        DataSet Ds = new DataSet();
        cn.Open();
        daProducts.Fill(Ds, "Products");
        cn.Close();
        cn = null;
        daProducts = null;
        return Ds;
}
```

13. Build and save the project
14. Open a browser and navigate to: *http://<URL>/Service.asmx*
    (replace <URL> with your Web service URL)
15. You should see the two web methods created above:

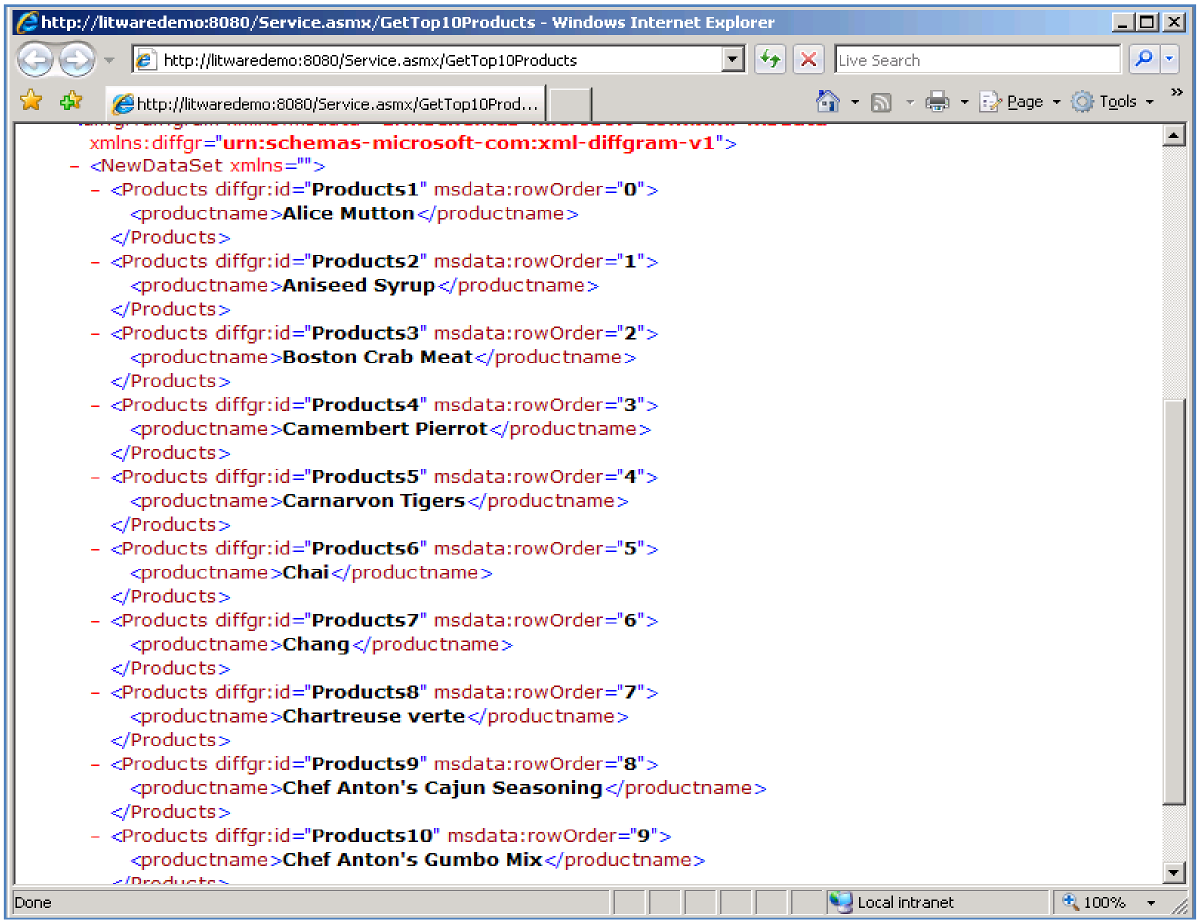


Service

The following operations are supported. For a formal definition, please review the Service Description.

- GetProductsBySupplierID
- GetTop10Products

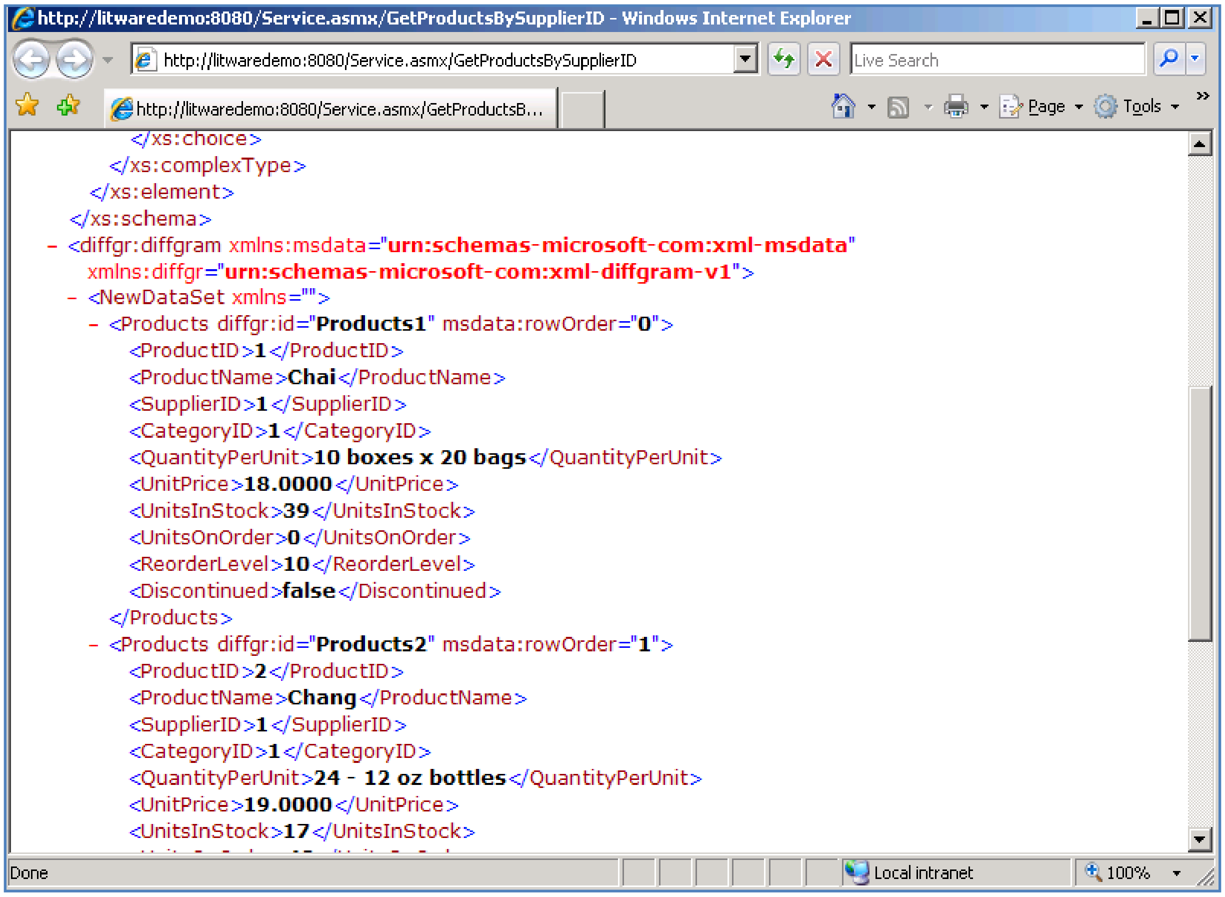
16. Click the *GetTop10Products* link and then click Invoke – this should return a list of the *products*



http://litwaredemo:8080/Service.asmx/GetTop10Products - Windows Internet Explorer
http://litwaredemo:8080/Service.asmx/GetTop10Products
Live Search
http://litwaredemo:8080/Service.asmx/GetTop10Prod...
Page
Tools
Done
Local intranet
100%

```
   xmlns:diffgr="urn:schemas-microsoft-com:xml-diffgram-v1">
 - <NewDataSet xmlns="">
   - <Products diffgr:id="Products1" msdata:rowOrder="0">
       <productname>Alice Mutton</productname>
     </Products>
   - <Products diffgr:id="Products2" msdata:rowOrder="1">
       <productname>Aniseed Syrup</productname>
     </Products>
   - <Products diffgr:id="Products3" msdata:rowOrder="2">
       <productname>Boston Crab Meat</productname>
     </Products>
   - <Products diffgr:id="Products4" msdata:rowOrder="3">
       <productname>Camembert Pierrot</productname>
     </Products>
   - <Products diffgr:id="Products5" msdata:rowOrder="4">
       <productname>Carnarvon Tigers</productname>
     </Products>
   - <Products diffgr:id="Products6" msdata:rowOrder="5">
       <productname>Chai</productname>
     </Products>
   - <Products diffgr:id="Products7" msdata:rowOrder="6">
       <productname>Chang</productname>
     </Products>
   - <Products diffgr:id="Products8" msdata:rowOrder="7">
       <productname>Chartreuse verte</productname>
     </Products>
   - <Products diffgr:id="Products9" msdata:rowOrder="8">
       <productname>Chef Anton's Cajun Seasoning</productname>
     </Products>
   - <Products diffgr:id="Products10" msdata:rowOrder="9">
       <productname>Chef Anton's Gumbo Mix</productname>
```

17. Click the *GetProductsBySupplierID* link, in the *suppID* box enter: *1* and then click Invoke – this should return a list of those *products* whose supplierId =1


http://litwaredemo:8080/Service.asmx/GetProductsBySupplierID - Windows Internet Explorer
http://litwaredemo:8080/Service.asmx/GetProductsBySupplierID
Live Search
http://litwaredemo:8080/Service.asmx/GetProductsB...
Page
Tools
Done
Local intranet
100%

```
            </xs:choice>
          </xs:complexType>
        </xs:element>
      </xs:schema>
  - <diffgr:diffgram xmlns:msdata="urn:schemas-microsoft-com:xml-msdata"
      xmlns:diffgr="urn:schemas-microsoft-com:xml-diffgram-v1">
    - <NewDataSet xmlns="">
      - <Products diffgr:id="Products1" msdata:rowOrder="0">
          <ProductID>1</ProductID>
          <ProductName>Chai</ProductName>
          <SupplierID>1</SupplierID>
          <CategoryID>1</CategoryID>
          <QuantityPerUnit>10 boxes x 20 bags</QuantityPerUnit>
          <UnitPrice>18.0000</UnitPrice>
          <UnitsInStock>39</UnitsInStock>
          <UnitsOnOrder>0</UnitsOnOrder>
          <ReorderLevel>10</ReorderLevel>
          <Discontinued>false</Discontinued>
        </Products>
      - <Products diffgr:id="Products2" msdata:rowOrder="1">
          <ProductID>2</ProductID>
          <ProductName>Chang</ProductName>
          <SupplierID>1</SupplierID>
          <CategoryID>1</CategoryID>
          <QuantityPerUnit>24 - 12 oz bottles</QuantityPerUnit>
          <UnitPrice>19.0000</UnitPrice>
          <UnitsInStock>17</UnitsInStock>
```

Exposing ProductsBySupplierID stored proc as a document literal SOAP Web Service:

With our ASMX web service completely coded, let's move on to creating the SOAP endpoint in SQL Server. Since the release SQL Server 2005 , a new feature has been added to this product that allows you to create SOAP “endpoints” which can be used to expose a stored procedure as a document literal SOAP Web Service.

Simply think of it as a wrapper around a stored procedure over HTTP. To set up SOAP endpoints in SQL Server 2005, you can use CREATE ENDPOINT T-SQL. For more information on creating endpoints in SQL Server 2005, see http://msdn2.microsoft.com/en-gb/library/ms345123.aspx .

Important
The SOAP endpoint that we are going to create ,in just a moment, will come handy in the upcoming blog post, when we create Full trust, master-detail and browser-enabled InfoPath forms as a composite no-code solution.

In order to create this endpoint, follow these steps:

1. Start SQL Server Management Studio

2. Create the following stored procedure in Northwind database

```
CREATE PROCEDURE [dbo].[ProductsBySupplierID] @SupID int

AS
SELECT ProductName
FROM Products
WHERE SupplierID = @SupID
```

3. Paste the following TSQL statement in a New Query window and execute it:

```
CREATE ENDPOINT getProductsBySupplierEndpoint
STATE = STARTED
AS HTTP
(
  SITE = 'LITWAREDEMO',
  PATH = '/getproductsbysupplierwebservice',
  AUTHENTICATION = ( NTLM ),
  PORTS = ( CLEAR )
)
FOR SOAP
(
  WEBMETHOD 'GetProductsBySupplier'
  (
    NAME = 'Northwind.dbo.ProductsBySupplierID',
    SCHEMA = DEFAULT,
    FORMAT = ROWSETS_ONLY
  ),
  WSDL = DEFAULT,
  BATCHES = DISABLED,
  DATABASE = 'Northwind'
)
```

The above TSQL syntax to create a SAOP endpoint contains three major parts:

A) The first part starts with CREATE and ends before the AS clause.

```
CREATE ENDPOINT getProductsBySupplierEndpoint

STATE = STARTED
```

In this case, endpoint name is set to getProductsBySupplierEndpoint. STARTED means that endpoint is started and is actively listening for connections.

B) The second part starts with AS and ends before the FOR clause.

```
AS HTTP
(
  SITE = 'LITWAREDEMO',
  PATH = '/getproductsbysupplierwebservice',
  AUTHENTICATION = ( NTLM ),
  PORTS = ( CLEAR )
)
```

In this case, HTTP is chosen as the transport protocol and incoming requests must use HTTP .The method of endpoint authentication is set to NTLM (Watch out for double-hop issue!) and PATH identifies the location of the endpoint on the host computer specified in the SITE argument.

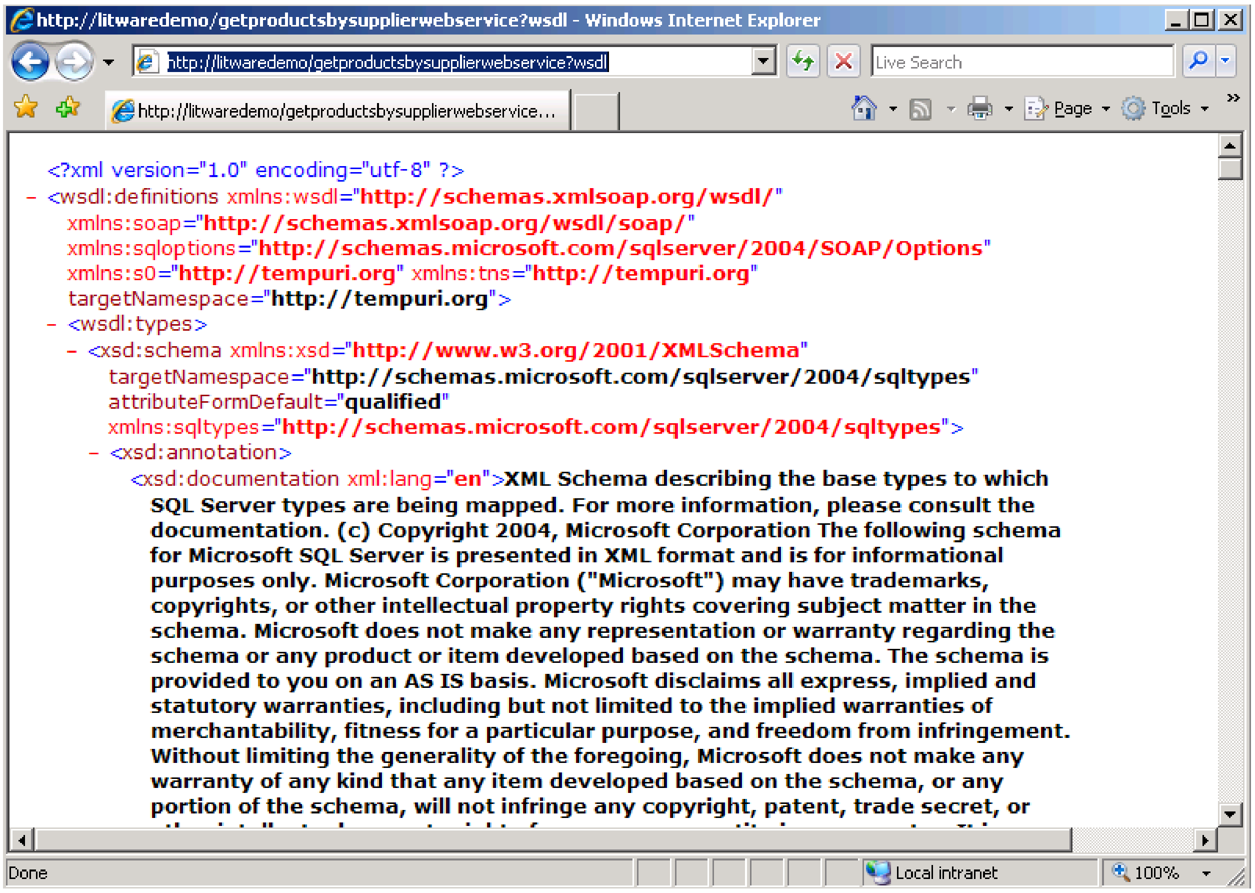
C) The third part starts with the FOR clause.

```
FOR SOAP
(
  WEBMETHOD 'GetProductsBySupplier'
  (
    NAME = 'Northwind.dbo.ProductsBySupplierID',
    SCHEMA = DEFAULT,
    FORMAT = ROWSETS_ONLY
  ),
  WSDL = DEFAULT,
  BATCHES = DISABLED,
  DATABASE = 'Northwind'
)
```

In this case, the payload that is supported on the endpoint is identified as SOAP. ProductsBySupplierID stored procedure which we created above is also specified to be exposed in the endpoint as a Web method (this also can be a user-defined function). Ad-hoc SQL requests in disabled to this endpoint by default (BATCHES = DISABLED). Since we are going to use the Visual Studio dynamic proxy class generator, we would like the results to be returned as a single dataset (System.Data.Dataset object) and not as an object array , so we set FORMAT to ROWSETS_ONLY. Other arguments are self-explanatory so I will skip drilling into them.

4. Open a browser and navigate to: *http://<SITE>/getproductsbysupplierwebservice?wsdl* (<SITE> with the name you chose for SITE argument in the above TSQL statement. I chose LITWAREDEMO in this case)

5. You should see the following result being returned from your Web service:


http://litwaredemo/getproductsbysupplierwebservice?wsdl - Windows Internet Explorer
http://litwaredemo/getproductsbysupplierwebservice?wsdl
Live Search
http://litwaredemo/getproductsbysupplierwebservice...
Page
Tools
Done
Local intranet
100%

```
  <?xml version="1.0" encoding="utf-8" ?>
- <wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    xmlns:sqloptions="http://schemas.microsoft.com/sqlserver/2004/SOAP/Options"
    xmlns:s0="http://tempuri.org" xmlns:tns="http://tempuri.org"
    targetNamespace="http://tempuri.org">
  - <wsdl:types>
    - <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        targetNamespace="http://schemas.microsoft.com/sqlserver/2004/sqltypes"
        attributeFormDefault="qualified"
        xmlns:sqltypes="http://schemas.microsoft.com/sqlserver/2004/sqltypes">
      - <xsd:annotation>
          <xsd:documentation xml:lang="en">XML Schema describing the base types to which
            SQL Server types are being mapped. For more information, please consult the
            documentation. (c) Copyright 2004, Microsoft Corporation The following schema
            for Microsoft SQL Server is presented in XML format and is for informational
            purposes only. Microsoft Corporation ("Microsoft") may have trademarks,
            copyrights, or other intellectual property rights covering subject matter in the
            schema. Microsoft does not make any representation or warranty regarding the
            schema or any product or item developed based on the schema. The schema is
            provided to you on an AS IS basis. Microsoft disclaims all express, implied and
            statutory warranties, including but not limited to the implied warranties of
            merchantability, fitness for a particular purpose, and freedom from infringement.
            Without limiting the generality of the foregoing, Microsoft does not make any
            warranty of any kind that any item developed based on the schema, or any
            portion of the schema, will not infringe any copyright, patent, trade secret, or
```

Now that we have coded both soap endpoints, let's move on to the next part. In Part 3, I will be walking you through InfoPath forms and how they can be leveraged to interact with LOB data. In particular, we will be looking into Browser-Enabled InfoPath forms.