How to Consume Enterprise Services with Microsoft .NET 3.0 and Visual Studio 2005

Applies to:
Access to the ES-Workplace mySAP ERP ECC 600.
Microsoft .NET Framework 3.0.

Summary
This document enables you to generate a Windows Communication Foundation Client for consuming Web Services from the ES Workplace using Visual Studio 2005 and the .NET Framework 3.0.

Time requirement
The time requirement for this How-To-Guide is about 20 minutes.

Author: Boris Mueller
Company: SAP AG
Created on: 20 February 2007

Author Bio
Boris Mueller is computer science student and currently works as a trainee at SAP in the Market Development Engineering Team in Walldorf.
Boris has several years of experience as maintenance supervisor of electrical engineering and telecommunications in a management position.
Table of Contents

Applies to: ........................................................................................................................................ 1
Summary.......................................................................................................................................... 1
Time requirement ............................................................................................................................. 1
Author Bio ........................................................................................................................................ 1

Part 1: Creating a project .................................................................................................................. 3
  Creating a Console Application .................................................................................................... 3
    Use ........................................................................................................................................... 3
    Procedure ................................................................................................................................. 3

Part 2: Choose a WSDL file from the ES Workplace ....................................................................... 4
  Choose a WSDL file ..................................................................................................................... 4
    Use ........................................................................................................................................... 4
    Procedure ................................................................................................................................. 4

Part 3: Generating WCF Source Code with the ServiceModel Metadata Utility Tool ...................... 7
  Rename the Message Part in the WSDL ..................................................................................... 7
    Use ........................................................................................................................................... 7
    Procedure ................................................................................................................................. 7
  Generating WCF Source Code with Svcutil ................................................................................. 7
    Use ........................................................................................................................................... 7
    Procedure ................................................................................................................................. 7

Part 4: Importing the Source Code and Consuming the Enterprise Service ................................... 9
  Importing the Source Code .......................................................................................................... 9
    Use ........................................................................................................................................... 9
    Procedure ................................................................................................................................. 9
  Creating Source Code ................................................................................................................ 10
    Use ........................................................................................................................................... 10
    Procedure ................................................................................................................................. 10
  Settings for the App.config file ................................................................................................... 11
    Use ........................................................................................................................................... 11
    Procedure ................................................................................................................................. 11
  Build the Project and Start the Program .................................................................................... 12

Complete Source Code .................................................................................................................. 13

Related Content............................................................................................................................. 16

Copyright........................................................................................................................................ 17
Part 1: Creating a project

Creating a Console Application

Use

First, we create a simple Console Application project in Visual Studio 2005. In part four, we include the generated WCF source code in this project.

Procedure

2. Choose File → New → Project
3. In the pop-up window, choose Visual C# → Console Application.
4. Enter the project name: WsCall
5. Choose OK to create a new Console Application.
Part 2: Choose a WSDL file from the ES Workplace

Choose a WSDL file

Use

The ServiceModel Metadata Utility Tool needs a WSDL file to generate a source code. In the following step, we will use the “Find Customer by Address” enterprise service, which you can find on the ES Workplace. The Name of this Enterprise Service in the mySAP ERP ECC 600 backend system is ECC_CUSTOMER002QR.

Procedure

1. Open the ES Workplace: http://erp.esworkplace.sap.com/socoview
2. Click the link Enterprise Service Index.
3. In the browser window, insert the name of the enterprise service “Find Customer by Address” into the field Search and Click Search in Enterprise Service Index.
4. The search results will appear in your browser window.
5. Click the link Find Customer by Address.
6. In the resulting page, you can get detailed information about the enterprise service.
7. To open the WSDL file, click the link on Related Web Service Definition.

**Find Customer by Address**

*Enterprise Service Operation*

<table>
<thead>
<tr>
<th>Technical Data</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Entity Type</strong></td>
<td>Service Operation</td>
</tr>
<tr>
<td><strong>Technical Name in ESR</strong></td>
<td>CustomerSimpleByAddressQueryResponse_In</td>
</tr>
<tr>
<td><strong>Namespace in ESR</strong></td>
<td><a href="http://sap.com/xml/APPL/5E/Global">http://sap.com/xml/APPL/5E/Global</a></td>
</tr>
<tr>
<td><strong>Software Component Version in ESR</strong></td>
<td>As of ECC-5E 600 SP00/01</td>
</tr>
<tr>
<td><strong>Category</strong></td>
<td>Inbound</td>
</tr>
<tr>
<td><strong>Mode</strong></td>
<td>Synchronous</td>
</tr>
<tr>
<td><strong>Related Web Service Definition</strong></td>
<td>ECC_CUSTOMER0020R</td>
</tr>
<tr>
<td><strong>Message Type Request</strong></td>
<td>Customer Simple By Address Query</td>
</tr>
<tr>
<td><strong>Message Type Response</strong></td>
<td>Customer Simple By Address Response</td>
</tr>
</tbody>
</table>

- Click here to get detailed field description
8. When prompted, logon with your user.
9. Choose File → Save As… in the browser window.

10. Change the file extension from XML to WSDL and choose Save to save the WSDL file on your computer.
Part 3: Generating WCF Source Code with the ServiceModel Metadata Utility Tool

Rename the Message Part in the WSDL

Use
Before we use the ServiceModel Metadata Utility Tool (Svcutil), we have to change the name of the message part in the WSDL from "parameters" into "parameter". If we do not, Svcutil will generate incorrect source code.

Procedure
1. Open the WSDL file in a text editor.
2. Search for the message part and change the name from "parameters" into "parameter" (two times in this case).
3. Save the new WSDL file.

Generating WCF Source Code with Svcutil

Use
With a WSDL file, we are able to generate WCF source code with Svcutil. Svcutil can be found at the Windows SDK installation location: C:\Program Files\Microsoft SDKs\Windows\v6.0\Bin. You can find a detailed documentation of the tool at: http://msdn2.microsoft.com/en-us/library/aa347733.aspx.

Procedure
1. Open the command prompt.
2. Specify the syntax:
   `svcutil.exe /ser:Auto /d:n:\targetFolder /config:App.config n:\sourceFolder\ECC_CUSTOMER002QR.wsdl`
The following table shows the used options for the tool.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/directory:&lt;directory&gt;</td>
<td>Directory to create files in. Default: The current directory. Short form: /d:&lt;directory&gt;</td>
</tr>
<tr>
<td>/serializer:Auto</td>
<td>Automatically selects the serializer. This uses the Data Contract serializer. If this fails, the XmlSerializer is used. Short form: /ser:Auto</td>
</tr>
<tr>
<td>/config:&lt;configFile&gt;</td>
<td>Specifies the filename for the generated configuration file. Default: output.config</td>
</tr>
</tbody>
</table>

3. Press Enter and the Svcutil tool will generate a C# file and a config file.
Part 4: Importing the Source Code and Consuming the Enterprise Service

Importing the Source Code

Use
Before we can use the generated source code, we have to import it into the Console Application project.

Procedure
1. Right-click on the WsCall project and choose Add → Existing Item….
2. Choose the directory where the files reside.
3. In the popup window, choose Files of type: All Files (*.*)
4. Select the two generated files.
5. Click Add to finish importing the two files into the Console Application project.
Creating Source Code

Use

To invoke the Web service, you have to specify the proxy settings, authentications, and values of the request and response. Here, we use the Program class. The complete source code is also available at the end of this chapter.

Procedure

1. Open the class Program

2. Insert the using directives.

```csharp
using System;
using System.Collections.Generic;
using System.Text;
using System.Net;
```

3. Handle errors and exceptions with a try/catch statement in the main method.

```csharp
try
{
}
catch (WebException exp)
{
    System.Console.WriteLine(exp.ToString());
}
```

4. Create instances in the try-block.

```csharp
System.Console.WriteLine("Starting SOAP WebService Call\n");
CustomerSimpleByNameAndAddressQueryResponse_InClient service = new
CustomerSimpleByNameAndAddressQueryResponse_InClient();
OrganisationName oname = new OrganisationName();
CustomerSimpleByNameAndAddressQueryMessage_sync query = new
CustomerSimpleByNameAndAddressQueryMessage_sync();
CustomerSimpleByNameAndAddressQueryMessageSyncCustomerSimpleSelectionByNameAndAddress querydata = new
CustomerSimpleByNameAndAddressQueryMessageSyncCustomerSimpleSelectionByNameAndAddress();
CustomerSimpleByNameAndAddressQueryMessageSyncCustomerSimpleSelectionByNameAndAddress querydata = new
CustomerSimpleByNameAndAddressQueryMessageSyncCustomerSimpleSelectionByNameAndAddress();
OrganisationName oname = new OrganisationName();
```
5. Set values for the request.

```csharp
oname.FirstLineName = "Julia";
querydata.CustomerAddressCountryCode = "DE";
query.CustomerSimpleSelectionByNameAndAddress = querydata;
querydata.CustomerName = oname;
```

6. Insert authentication and specify your username and password from the ES Workplace.

```csharp
service.ClientCredentials.UserName.UserName = "myUser";
service.ClientCredentials.UserName.Password = "myPassword";
```

6. Display the response.

```csharp
CustomerSimpleByNameAndAddressResponseMessage_sync results =
    service.CustomerSimpleByNameAndAddressQueryResponse_In(query);
CustomerSimpleByNameAndAddressResponseMessage_syncCustomer[] customers =
    results.Customer;
for (int i = 0; i < customers.Length; ++i)
{
    CustomerSimpleByNameAndAddressResponseMessage_syncCustomer cur = customers[i];
    if (i == 0) System.Console.WriteLine("Customer(s) found: "+ customers.Length + 
"\n");
    System.Console.WriteLine("\nCustomerID: " + cur.ID.Value);
"\n");
}
System.Console.WriteLine("\nWebServiceCall finished");
```

### Settings for the App.config file

**Use**

In the App.config file we can change the application settings.

**Procedure**

1. Open the App.config file.
2. Insert the proxy settings where they are necessary.

```xml
<system.diagnostics>
    <trace>
        < traceListeners >
            <aspnetTraceListener />
            <nlogTraceListener />
        </traceListeners>
    </trace>
</system.diagnostics>
```

3. Change the settings for security mode from “None” to “TransportCredentialOnly”
4. Change the settings for clientCredentialType from “None” to “Basic”.

```xml
<configuration>
    <system.webServices>
        <client>
            <endpoint address="http://myProxy.myCorp.com"
            binding="wsHttpBinding"
            bindingConfiguration="wsHttpBinding">
                <transport useDefaultWebProxy="false" />
            </endpoint>
        </client>
    </system.webServices>
</configuration>
```

5. Choose File → Save.
6. Right-click on References and choose Add Reference from the context menu.
7. In the popup window, add the following references:
   - System.Runtime.Serialization
   - System.ServiceModel

![](image)

**Build the Project and Start the Program**

1. Choose **Build → Build Solution** to build the project.

![](image)
2. Choose **Debug → Start Without Debugging** to start the program, then watch as the result of the Web service call is displayed in the console.

![Image](image.png)

**Complete Source Code**

The following is the complete source code for the config file and the program file. You can copy and paste it into the files. Before building the solution, be sure to change the settings for username “yourUser” and password “yourPassword” in the source code.

```csharp
// source code for the Program file
using System;
using System.Collections.Generic;
using System.Text;
using System.Net;
namespace WsClient
{
    class Program
    {
        static void Main(string[] args)
        {
            try
            {
```
System.Console.WriteLine("Starting SOAP WebService Call\n");
CustomerSimpleByNameAndAddressQueryResponse_InClient service = new
CustomerSimpleByNameAndAddressQueryResponse_InClient();
OrganisationName oname = new OrganisationName();
CustomerSimpleByNameAndAddressQueryMessage_sync query = new
CustomerSimpleByNameAndAddressQueryMessage_sync();
CustomerSimpleByNameAndAddressQueryMessage_syncCustomerSimpleSelectionByNameAndAddress querydata = new
CustomerSimpleByNameAndAddressQueryMessage_syncCustomerSimpleSelectionByNameAndAddress();
oname.FirstLineName = "Julia";
querydata.CustomerAddressCountryCode = "DE";
query.CustomerSimpleSelectionByNameAndAddress = querydata;
querydata.CustomerName = oname;
service.ClientCredentials.UserName.UserName = "MyUser";
service.ClientCredentials.UserName.Password = "MyPassword";
CustomerSimpleByNameAndAddressResponseMessage_sync results =
service.CustomerSimpleByNameAndAddressQueryResponse_In(query);
CustomerSimpleByNameAndAddressResponseMessage_syncCustomer[] customers =
results.Customer;
for (int i = 0; i < customers.Length; ++i)
{
    CustomerSimpleByNameAndAddressResponseMessage_syncCustomer cur = customers[i];
    if (i == 0) System.Console.WriteLine("Customer(s) found: " + customers.Length + 
        "\n");
    System.Console.WriteLine("\nCustomerID: " + cur.ID.Value);
        "\n");
}
System.Console.WriteLine("\nWebServiceCall finished\n");
catch (WebException exp)
{
    System.Console.WriteLine(exp.ToString());
}]]>

<!-- source code for the config file -->
<?xml version="1.0" encoding="utf-8"?>
<configuration>
    <system.serviceModel>
        <bindings>
            <basicHttpBinding>
                <binding name="CustomerSimpleByNameAndAddressQueryResponse_InSoapBinding"
                    closeTimeout="00:01:00" openTimeout="00:01:00" receiveTimeout="00:10:00"
                    sendTimeout="00:01:00" allowCookies="false" bypassProxyOnLocal="false"
                    hostNameComparisonMode="StrongWildcard" maxBufferSize="65536"
                    maxBufferPoolSize="524288" maxReceivedMessageSize="65536"
                    messageEncoding="Text" textEncoding="utf-8" transferMode="Buffered"
                    proxyAddress="http://MyProxy.MyCorp.com"/>
useDefaultWebProxy="false">
<readerQuotas maxDepth="32" maxStringContentLength="8192" maxArrayLength="16384"
maxBytesPerRead="4096" maxNameTableCharCount="16384" />
<security mode="TransportCredentialOnly">
<transport clientCredentialType="Basic" proxyCredentialType="None" realm=""/>
<message clientCredentialType="UserName" algorithmSuite="Default" /></security>
</binding>
</basicHttpBinding>
</bindings>
</client>
<endpoint
address="http://erp.esworkplace.sap.com:80/sap/bc/srt/xip/sap/ECC_CUSTOMER002QR?
sap-client=800"
binding="basicHttpBinding"
bindingConfiguration="CustomerSimpleByNameAndAddressQueryResponse_InSoapBinding"
contract="CustomerSimpleByNameAndAddressQueryResponse_In"
name="CustomerSimpleByNameAndAddressQueryResponse_InSoapBinding" />
</client>
</system.serviceModel>
</configuration>
Related Content

- Platform Ecosystem
- ES Workplace homepage on SDN
- Enterprise SOA homepage on SDN
Copyright

© Copyright 2007 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors. Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, Informix, i5/OS, POWER, POWER5, OpenPower and PowerPC are trademarks or registered trademarks of IBM Corporation.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

MaxDB is a trademark of MySQL AB, Sweden.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

These materials are provided “as is” without a warranty of any kind, either express or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.

SAP shall not be liable for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials.

SAP does not warrant the accuracy or completeness of the information, text, graphics, links or other items contained within these materials. SAP has no control over the information that you may access through the use of hot links contained in these materials and does not endorse your use of third party web pages nor provide any warranty whatsoever relating to third party web pages.

Any software coding and/or code lines/strings (“Code”) included in this documentation are only examples and are not intended to be used in a productive system environment. The Code is only intended better explain and visualize the syntax and phrasing rules of certain coding. SAP does not warrant the correctness and completeness of the Code given herein, and SAP shall not be liable for errors or damages caused by the usage of the Code, except if such damages were caused by SAP intentionally or grossly negligent.