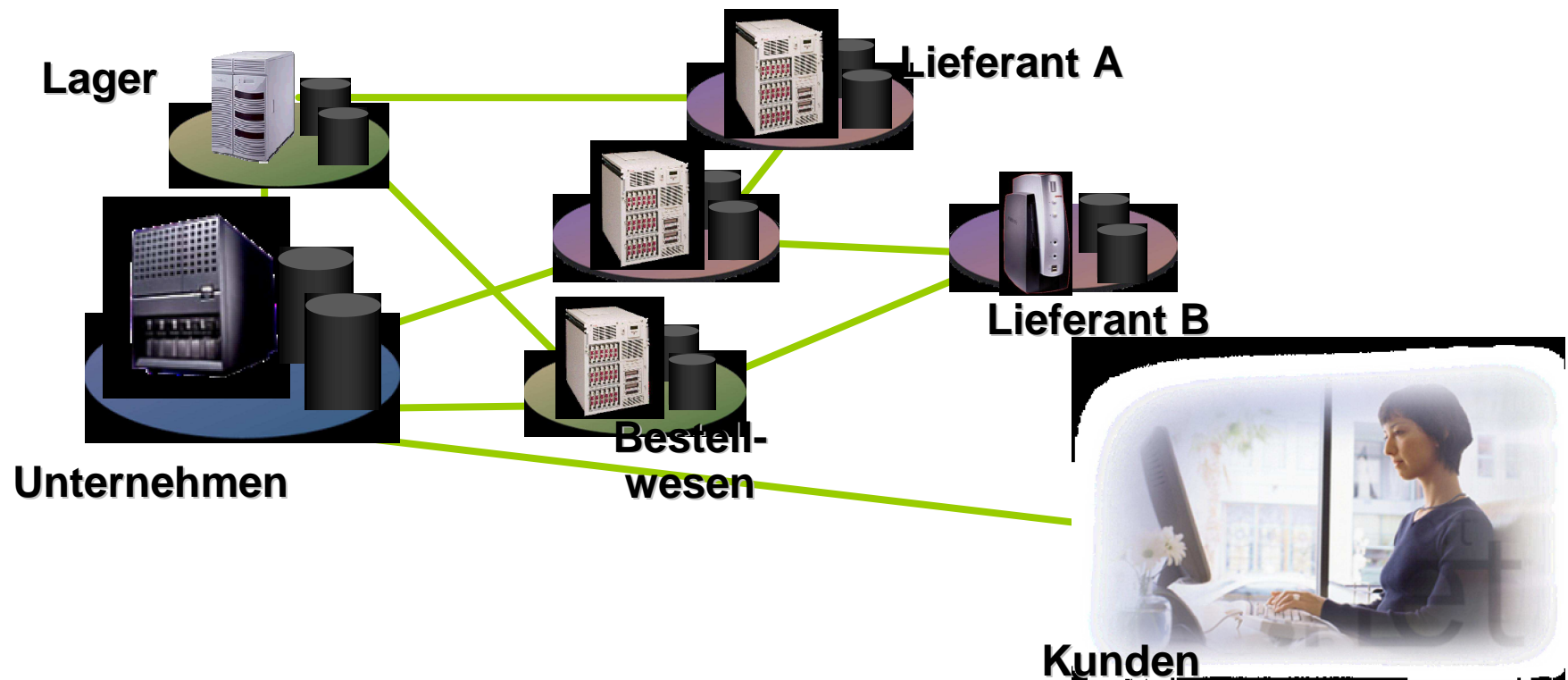


Webservices mit .NET

Dienste im Netz, die über Standard-Protokolle erreichbar sind
Datenaustausch basiert auf XML- Dokumenten
Plattform- und unternehmensübergreifende Transaktionen



Web Services bieten ...

- programmiersprachenunabhängige,
- protokollunabhängige,
- intelligente,
- kontextsensitive,
- XML- und webbasierte

Inter-Applikationskommunikation

Web Service Directory : UDDI

Beschreibung von WS : WSDL

Datenaustausch zw. WS : SOAP

Universelles Datenformat : XML

Kommunikationsnetz : Internet/HTTP

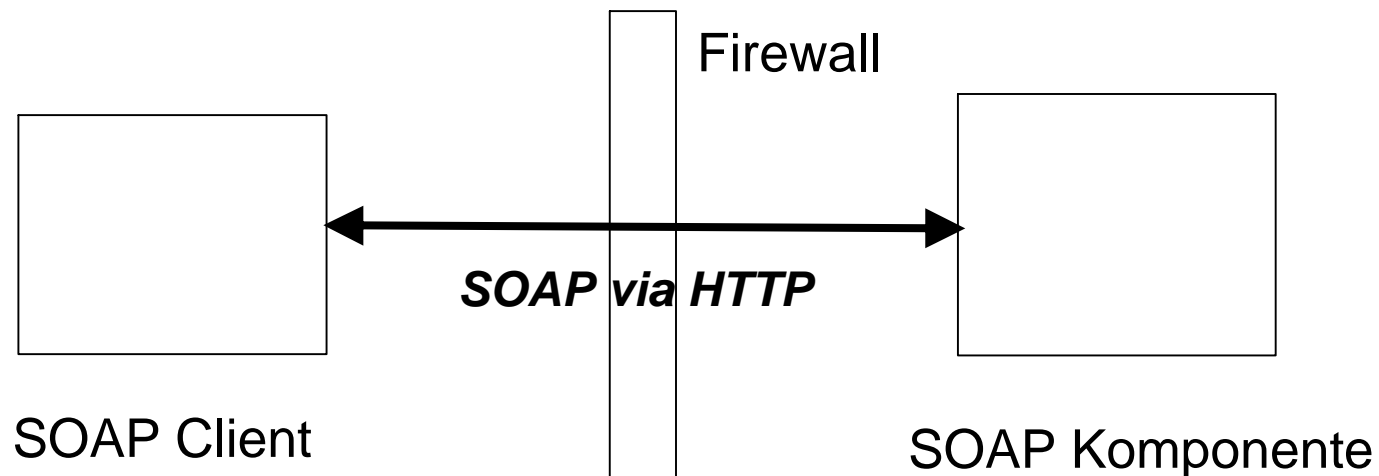
⇒ *einfache, offene und akzeptierte Standards*

Simple Object Access Protocol

- Lightweight Message Protokoll
- Typisierter Datenaustausch zwischen Applikationen
- Besonders geeignet für RPC Funktionalität
- Lose Kopplung
- Verteilte Architektur
- Transport: Binding über darunterliegendes Transportprotokoll
- Struktur wird in XML definiert

Problem :

- Wie können Komponenten oder Dienste auf fernen Rechnern problemlos angesprochen werden?
 - DCOM → Aufwendig und schwierig ; Stichwort : **Firewall**
 - SOAP → Einfacher (kein Firewallproblem)



Das Simple Object Access Protocol (SOAP)

- SOAP stellt einfachen, auf XML basierenden Mechanismus zum Austausch von strukturierten Informationen zwischen zwei Mitgliedern einer dezentralisierten, verteilten Umgebung zur Verfügung. (W3C)

Ziele von SOAP:

- Einfachheit und Erweiterbarkeit
- Definition eines Standardprotokolls für den Aufruf von Objekten
- Basis bilden Internetstandards : HTTP für Transport und XML für Datenkodierung

**SOAP ist Basistechnologie der
WebServices innerhalb von .NET !**

Ein einfacher SOAP – Request über HTTP:

```
POST /StockQuote HTTP/1.1
Host: www.stockquoteserver.com
Content-Type: text/xml; charset="utf-8"
SOAPAction: "Some-URI"
```

HTTP-Header

```
<SOAP:Envelope
  xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" >
  <SOAP:Body>
    <m:GetLastTradePrice xmlns:m="Some-URI">
      <symbol>DIS</symbol>
    </m:GetLastTradePrice>
  </SOAP:Body>
</SOAP:Envelope>
```

Rumpf

**Umschlag
(eigentliche SOAP-Nachricht)**

Ein einfacher SOAP – Response des Servers über HTTP :

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn
```

HTTP-Header

```
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-
  ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
  <SOAP-ENV:Body>
    <m:GetLastTradePriceResponse xmlns:m="Some-URI">
      <Price>34.5</Price>
    </m:GetLastTradePriceResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

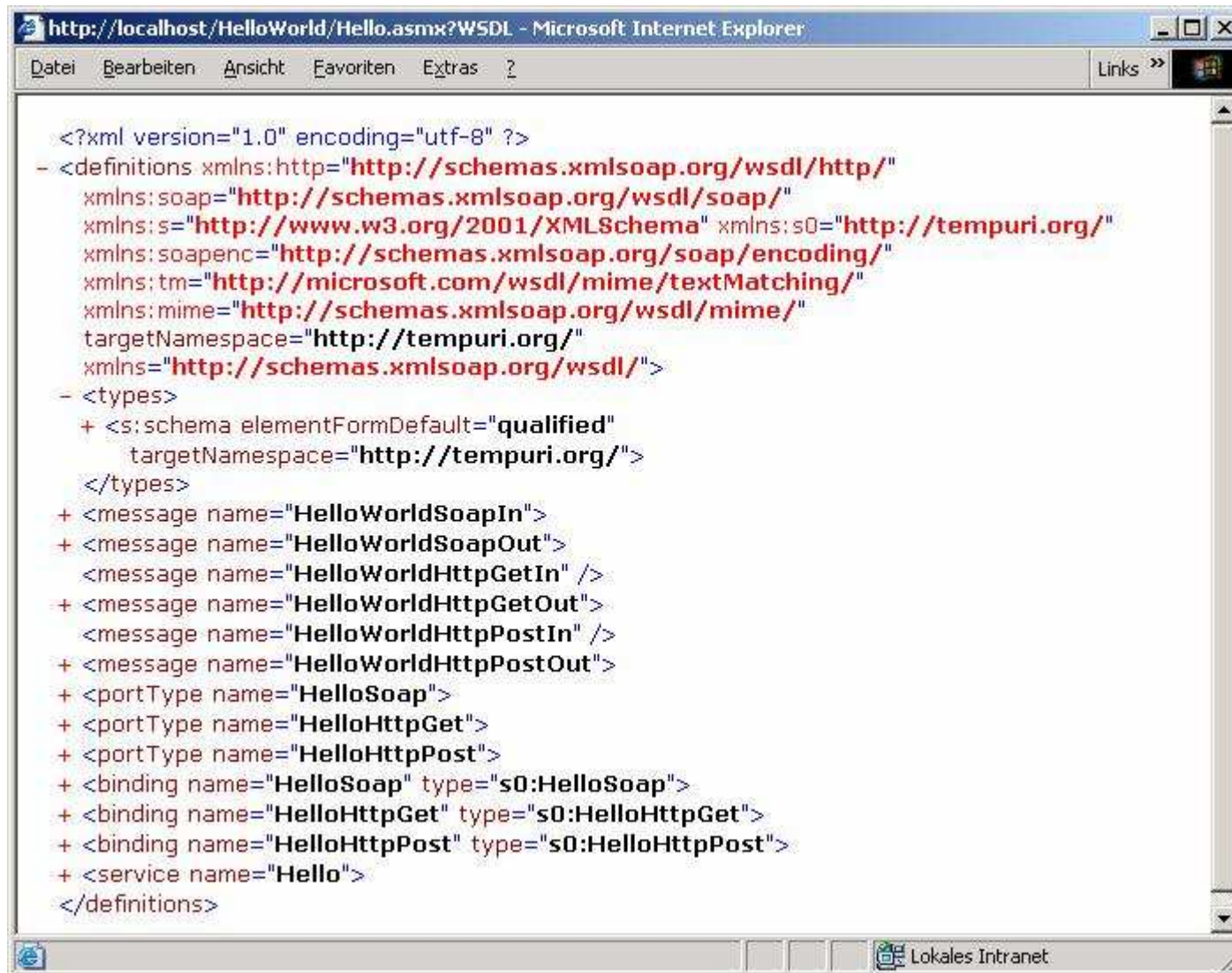
Umschlag

Rumpf

- Web Service Description Language
- W3C-Spezifikation (<http://www.w3.org/TR/wsdl>)
- XML basiert
- Beschreibung von Web-Services (wie IDL in CORBA)
- Kann aus Klassen gewonnen werden (und umgekehrt)

Gibt Antworten auf die Fragen:

1. Wo liegt der Web-Service genau?
2. Wie kann man mit ihm arbeiten, welche Schnittstellen hat er?
3. Mit welchen Protokollen kann man den Web-Service verwenden?



```
<?xml version="1.0" encoding="utf-8" ?>
- <definitions xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:s="http://www.w3.org/2001/XMLSchema" xmlns:s0="http://tempuri.org/"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
  xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
  targetNamespace="http://tempuri.org/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">
- <types>
+ <s:schema elementFormDefault="qualified"
  targetNamespace="http://tempuri.org/">
  </types>
+ <message name="HelloWorldSoapIn">
+ <message name="HelloWorldSoapOut">
  <message name="HelloWorldHttpGetIn" />
+ <message name="HelloWorldHttpGetOut">
  <message name="HelloWorldHttpPostIn" />
+ <message name="HelloWorldHttpPostOut">
+ <portType name="HelloSoap">
+ <portType name="HelloHttpGet">
+ <portType name="HelloHttpPost">
+ <binding name="HelloSoap" type="s0:HelloSoap">
+ <binding name="HelloHttpGet" type="s0:HelloHttpGet">
+ <binding name="HelloHttpPost" type="s0:HelloHttpPost">
+ <service name="Hello">
</definitions>
```

Web Services Description Tool innerhalb .NET

- Erzeugen von .NET Client Proxies
 - Komplette Proxyklasse (synchrone und asynchrone Aufrufe)
 - Notwendiger Code für Netzwerkkommunikation
- Quelldatei in angegebener Sprache

Verschiedene Eingabeparameter, u.a.

- URL einer WSDL Datei
- Implementierungssprache der Proxyklasse
- Aufruf-Protokoll
- Benutzername und Passwort bei Basic Authentication

Global und herstellerübergreifend

- UDDI
 - Universal Discovery, Description and Integration
 - Globales Web Services Verzeichnis

Lokal und proprietär (MS)

- DISCOvery “Protokoll” (Vorläufer von UDDI)
 - .vsdisco Dateien in Visual Studio .NET
 - myWS.asmx?disco
 - Mit disco.exe erzeugte Dateien
- XML Grammatik
 - Enthält Verweise zu WSDL Ressourcen
- Abgelegt im Server Rootverzeichnis

Universal Description, Discovery and Integration

Dient zur Registrierung von Web Services; zentrales Verzeichnis von Unternehmen und Web Services

UDDI API als Web Service nach aussen offengelegt

Globale Business-Registry:

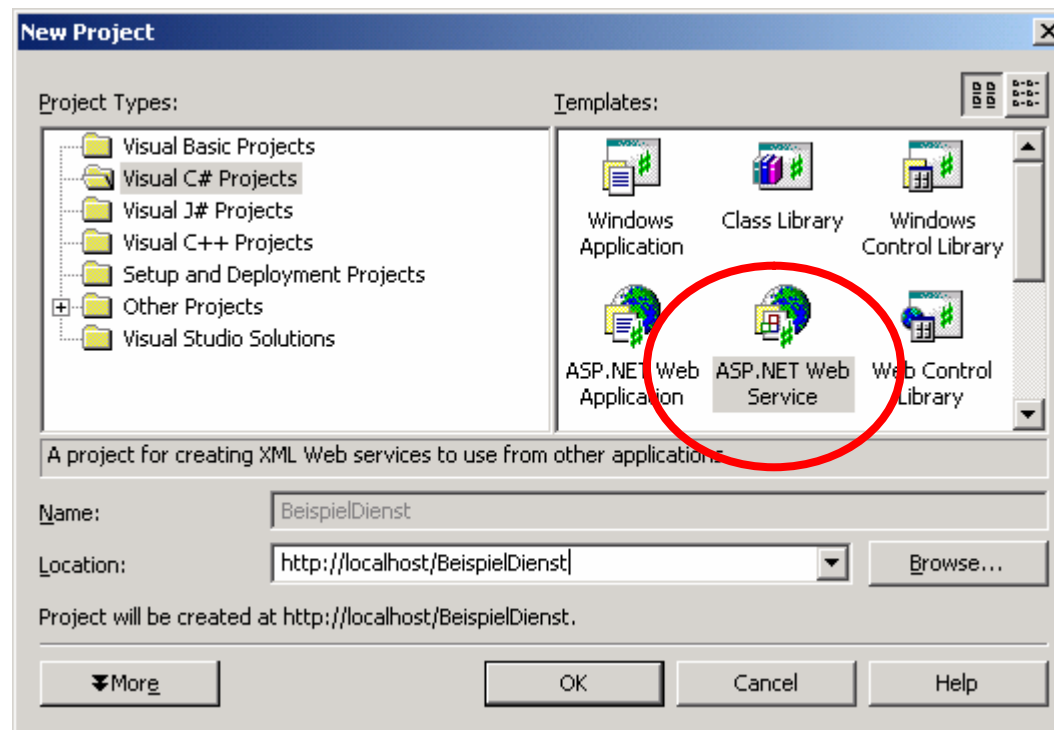
- Logisch zentral
- White-Pages:
 - Informationen zu den Anbietern (URL, Adresse, TelNr.)
- Yellow-Pages:
 - Branchenspezifische Suche nach Unternehmen (Branche, Land)
- Green-Pages:
 - Technische Beschreibung der Plattformen und Dienste eines Unternehmens

Weitere Informationen: <http://www.uddi.org>

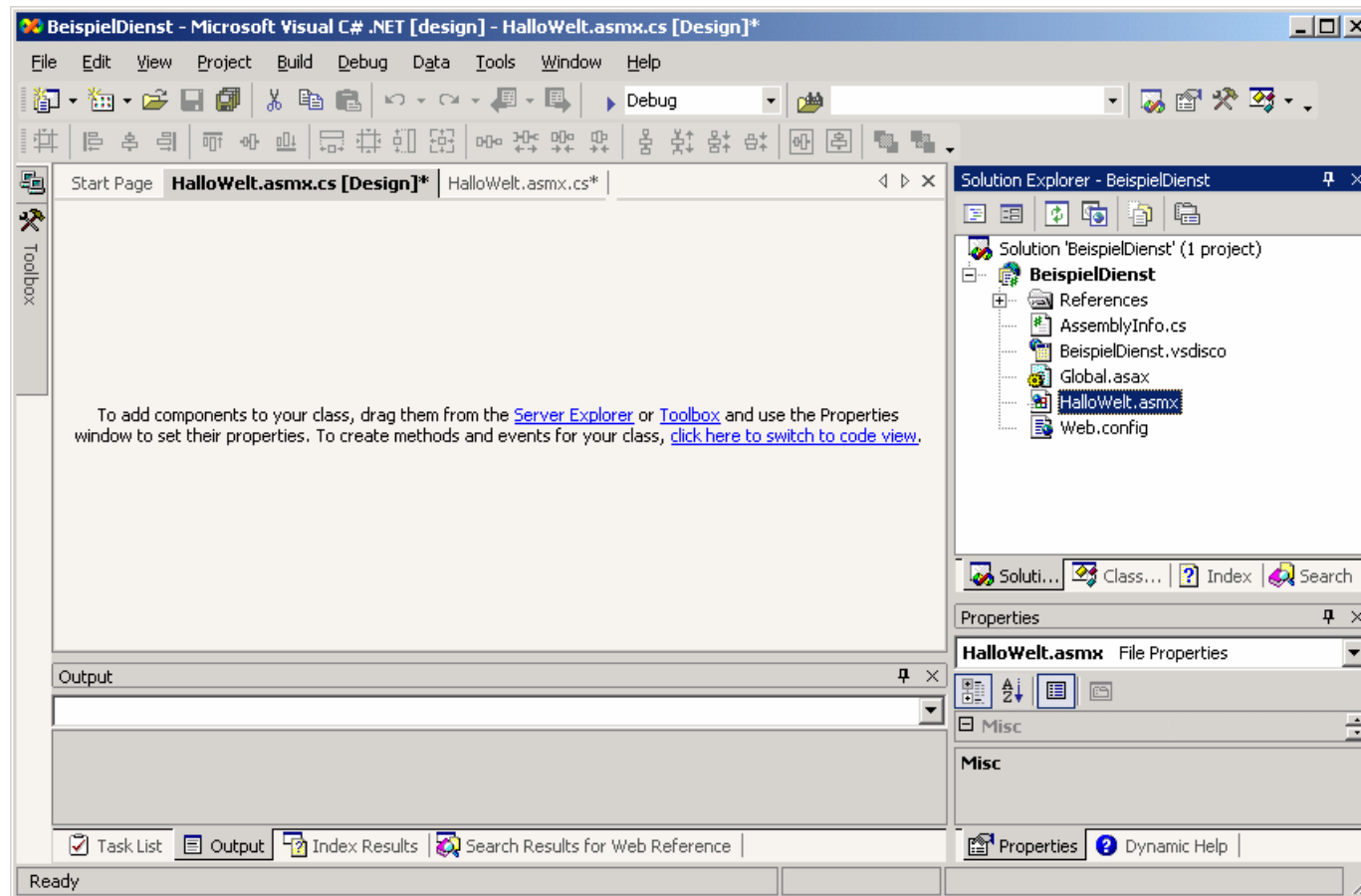
Webservices in .NET basieren auf offenen Standards

- Extensible Markup Language (XML)
 - grundlegender Datenbeschreibungsstandard auf der Präsentationsschicht
- Hypertext Transfer Protocol (HTTP)
 - Transportschicht-Standard
- Simple Object Access Protocol (SOAP)
 - Standardprotokoll zur Kommunikation zwischen Applikationen über das Web
- Web Services Description Language (WSDL)
 - Beschreibungssprache für Web Services

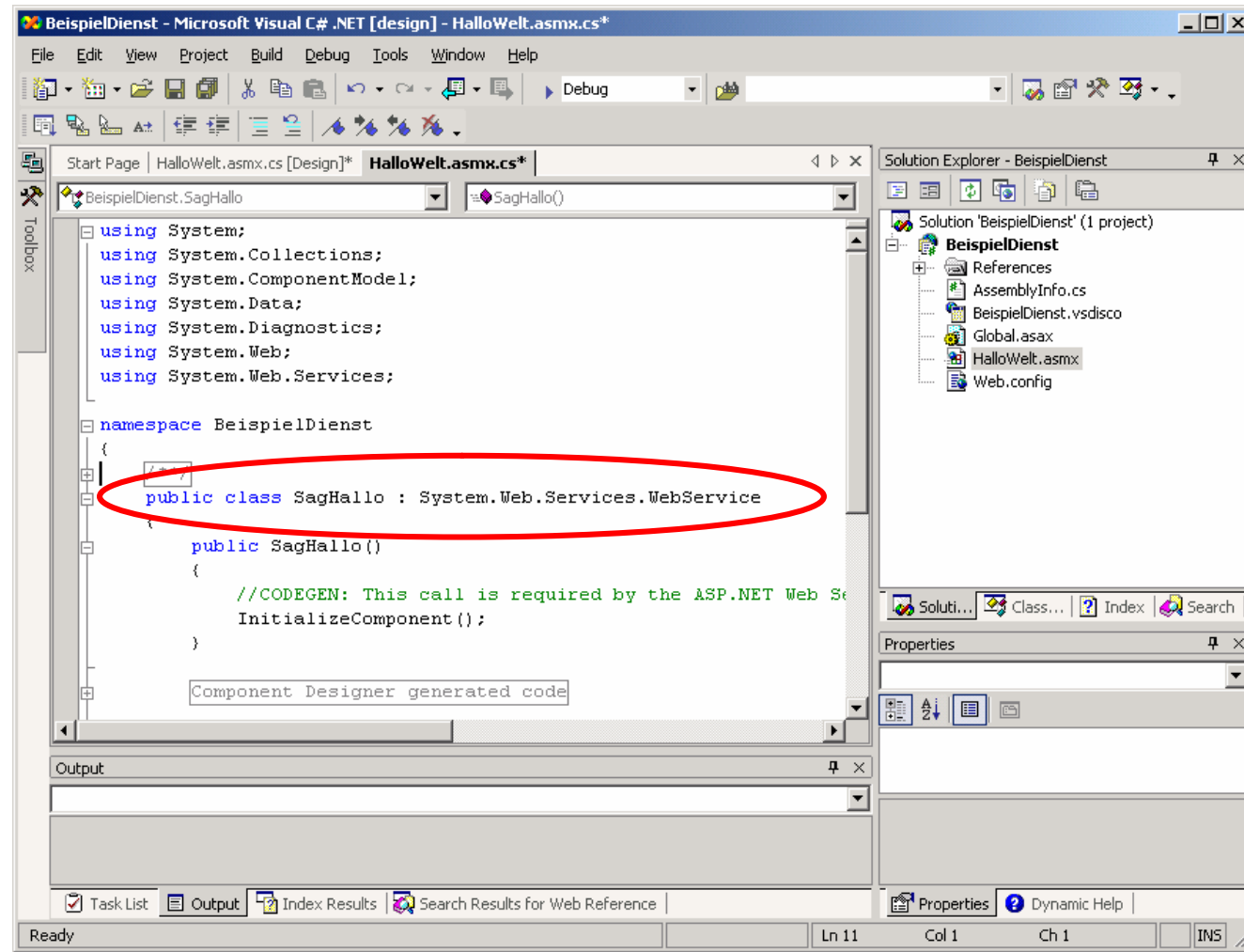
Erstellung eines einfachen Webservice mit Visual Studio.NET



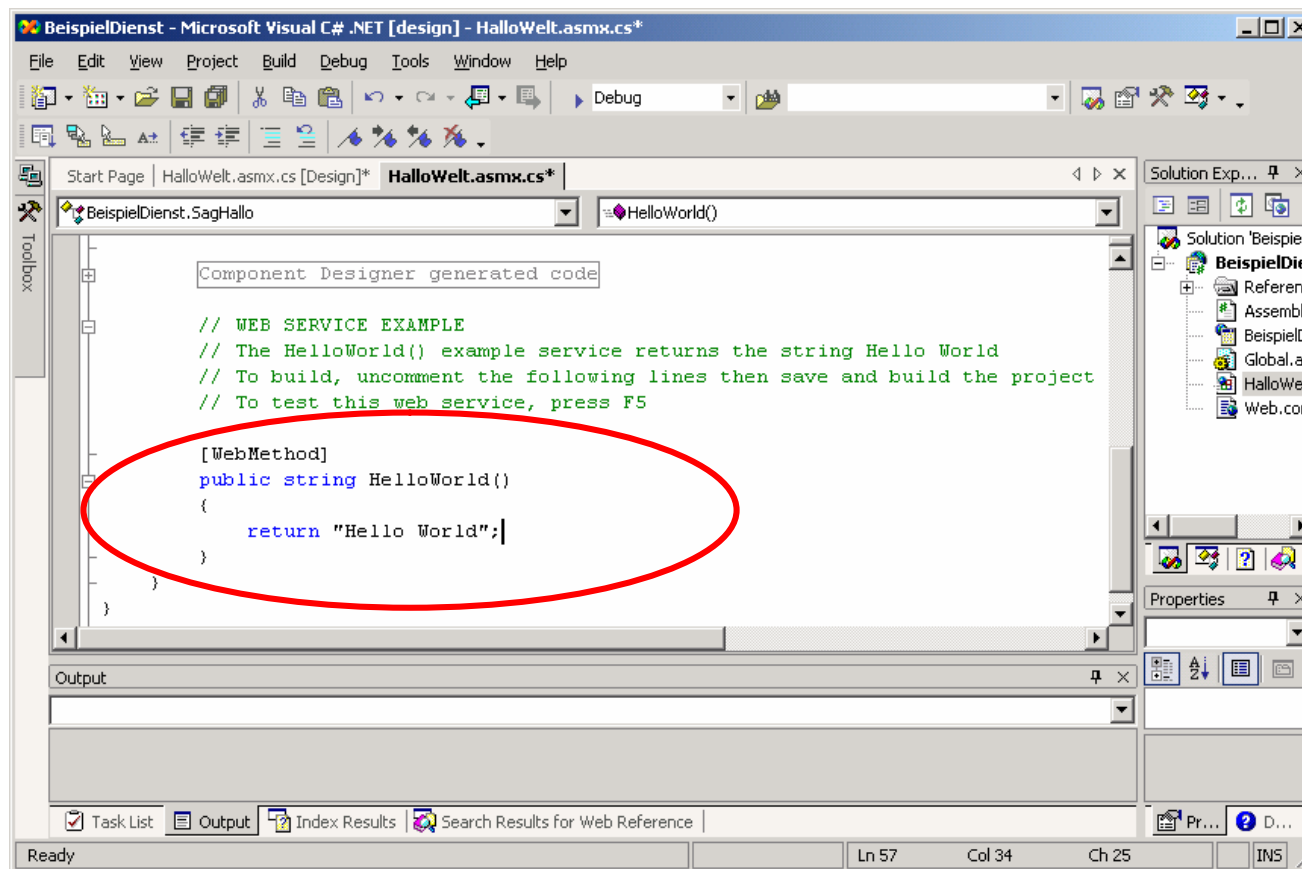
WebService Applikation :



WebService Applikation :



WebService Applikation :



Lokaler Test



WSDL - Datei

Lokaler Test



SagHallo Web Service - Microsoft Internet Explorer

Adresse <http://localhost/BeispielDienst/HalloWelt.aspx?op=HelloWorld>

HelloWorld

Test

To test the operation using the HTTP GET protocol, click the 'Invoke' button.

SOAP

The following is a sample SOAP request and response. The **placeholders** show need to be replaced with actual values.

```
POST /BeispielDienst/HalloWelt.aspx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/HelloWorld"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-
  <soap:Body>
    <HelloWorld xmlns="http://tempuri.org/" />
  </soap:Body>
</soap:Envelope>
```



http://localhost/BeispielDienst/HalloWelt.aspx/HelloWorld? - Microsoft...

Adresse <http://localhost/BeispielDienst/HalloWelt.aspx/HelloWorld?>

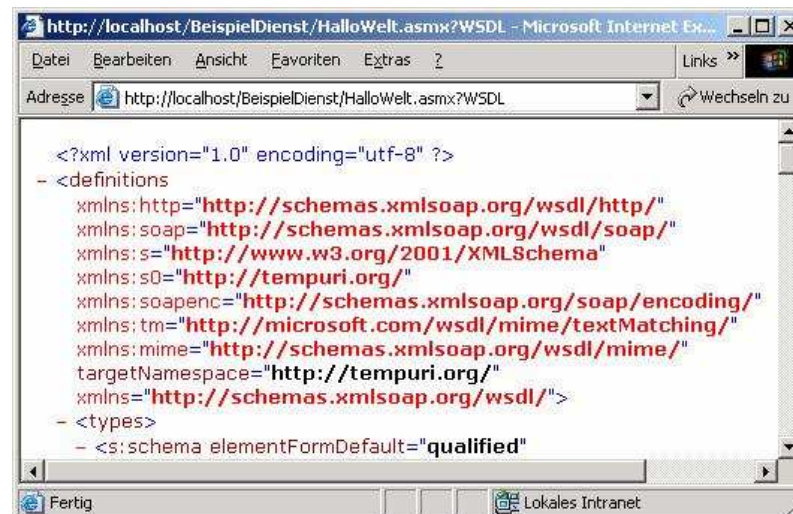
```
<?xml version="1.0" encoding="utf-8" ?>
<string xmlns="http://tempuri.org/">Hello World</string>
```

1. Schritt : Erstellung einer Proxyklasse unter Zuhilfenahme des .NET Tools „wsdl.exe“

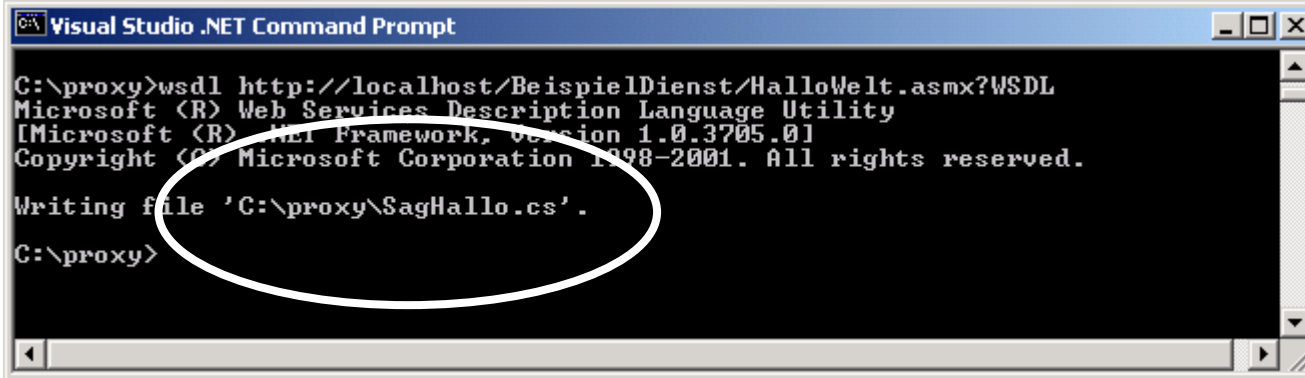
- Beschreibung des zu nutzenden Webdienstes ist erreichbar unter:

<http://localhost/BeispielDienst/HalloWelt.asmx?WSDL>

URL des Webdienstes



1. Schritt (Fortsetzung):



```
Visual Studio .NET Command Prompt
C:\proxy>wsdl http://localhost/BeispielDienst/HalloWelt.asmx?WSDL
Microsoft (R) Web Services Description Language Utility
[Microsoft (R) .NET Framework, Version 1.0.3705.01
Copyright (C) Microsoft Corporation 1998-2001. All rights reserved.
Writing file 'C:\proxy\SagHallo.cs'.
C:\proxy>
```

Übernahme dieser „Proxyklasse“ in ein aktuelles Projekt (C# Console Application)

2. Schritt

The screenshot shows the Visual Studio IDE with the 'Add Reference' dialog box open. The dialog box is titled 'Add Reference' and has tabs for '.NET', 'COM', and 'Projects'. The '.NET' tab is selected, and a list of components is displayed. The 'System.Web.Services.dll' component is highlighted in blue. The 'Selected Components' table is empty. The 'Class1.cs' file is open in the editor, and the 'using System.Web.Services;' statement is circled in red. The Solution Explorer shows the project structure, including the 'References' folder and the 'SagHallo.cs' file. An arrow points from the text 'Proxyklasse: SagHallo.cs' to the 'SagHallo.cs' file in the Solution Explorer.

Component Name	Version	Path
System.Messaging.dll	1.0.3300.0	C:\WINNT\Microsoft.NET\Fra...
System.Runtime.Remoting	1.0.3300.0	C:\WINNT\Microsoft.NET\Fra...
System.Runtime.Serialization...	1.0.3300.0	C:\WINNT\Microsoft.NET\Fra...
System.Security	1.0.3300.0	C:\WINNT\Microsoft.NET\Fra...
System.ServiceProcess.dll	1.0.3300.0	C:\WINNT\Microsoft.NET\Fra...
System.Web.dll	1.0.3300.0	C:\WINNT\Microsoft.NET\Fra...
System.Web.RegularExpressions...	1.0.3300.0	C:\WINNT\Microsoft.NET\Fra...
System.Web.Services.dll	1.0.3300.0	C:\WINNT\Microsoft.NET\Fra...
System.Windows.Forms.dll	1.0.3300.0	C:\WINNT\Microsoft.NET\Fra...
System.Xml.dll	1.0.3300.0	C:\WINNT\Microsoft.NET\Fra...
VJ5BrowserStubLib	1.0.3300.0	C:\WINNT\Microsoft Visual JS...
visocr	1.0.3300.0	C:\WINNT\Microsoft Visual JS...

Selected Components:

Component Name	Type	Source
----------------	------	--------

Proxyklasse: *SagHallo.cs*

3. Schritt : Nutzung des Webdienstes

