WEB Services

Einführung

Für den Betrieb von WEB Services wurde die sogenannte Service oriented Architektur (SOA) entwickelt und definiert. Diese beinhaltet Technologien und Werkzeuge, die es erlauben Daten und Prozesse über das HTTP Protokoll zu betreiben und auszutauschen. Den neuesten Trend in der Evolution der Applikationsarchitekturen stellen die Serviceorientierten Architekturen (kurz SOA) dar. Diese verteilten Architekturen basieren auf der Komposition loser gekoppelter Komponenten in Form von Diensten.

Durch die Repräsentation dieser Dienste durch Standards wie XML Web Services sind SOAs in den Brennpunkt der Softwarearchitekturszene gerückt.

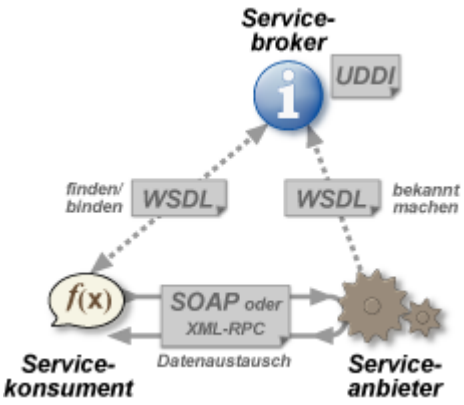
Architektur

Client-Programme senden im Allgemeinen Anfragen an einen Webservice, und dieser antwortet mit der gewünschten Information. Es wird daher gelegentlich behauptet, dass Webservices für Rechner das sind, was Webseiten für den Menschen sind. Auch wenn das nur einen Teil der Möglichkeiten der Webservices beschreibt, ist dieser Vergleich durchaus treffend. Webservices sind nicht für menschliche Benutzer gedacht, sondern für Softwaresysteme, die automatisiert Daten austauschen und/oder Funktionen auf entfernten Rechnern aufrufen.

Webservices orientieren sich an der Serviceorientierten Architektur (SOA) und vereinen daher verteilte und objektorientierte Programmierstandards und richten sich auf betriebswirtschaftliche Lösungen im Internet.

Es lassen sich die Instanzen Konsument, Anbieter und Verzeichnis identifizieren.

Der Anbieter veröffentlicht in einem Verzeichnis die Beschreibung seiner



Dienste. Der Konsument durchsucht das Verzeichnis und wählt den gewünschten Dienst aus. Nachdem eventuell weitere Protokolldetails ausgetauscht wurden, findet die dynamische Anbindung des Konsumenten an den Anbieter statt. Der Konsument greift nun auf Methoden zurück.

Die Grundlage hierbei bilden drei Standards, die jeweils auf XML basieren und in den zugehörigen Artikeln näher beschrieben werden:

- **UDDI** als Verzeichnisdienst zur Registrierung von Webservices. Es ermöglicht das dynamische Finden des Webservices (z. B. den Dienst FußballErgebnisse) durch den Konsumenten. Allerdings wird UDDI nur in eher kleineren Firmennetzwerken verwendet und hat sich nie global durchgesetzt.
- **WSDL** zur Beschreibung der unterstützten Methoden (z. B. TorschuetzenKoenig) und deren Parametern (z. B. Datum) für den Programmierer.
- **SOAP** (oder XML-RPC) zur Kommunikation. Hier wird der eigentliche Aufruf gestartet.

Webservices bilden die drei wichtigsten Teile der Zusammenarbeit zwischen Client und Server ab: das Zusammenfinden, Binden und den Datenaustausch.

Erreichbar sind Webservices über einen eindeutigen URI. Die verwendeten plattformunabhängigen Standards sind in der Lage, entfernte Methodenaufrufe beliebiger Plattformen zu dekodieren und an eine Anwendung weiterzuleiten. Auf diese Weise entsteht eine verteilte Architektur. Die Kommunikation mit Webservices erfolgt über Nachrichten, die über unterschiedliche Protokolle transportiert werden können.

Beispiele

Google Inc. betreibt seit 2002 einen Webservice, der durch seine Funktionalitäten die gleichen Möglichkeiten bietet wie die Benutzerschnittstelle auf der Google-Webseite selbst. Programme können nun mit einem Ansprechen der Schnittstelle direkt nach Informationen im Internet suchen, erhalten über die Schnittstelle die Ergebnisdaten und können diese für ihre eigenen Aufgaben verwenden. Das Parsen der Google-Webseite ist dazu keine auch nur annähernd gleichwertige Alternative. Allerdings stellt Google seit Dezember 2006 keine neuen Zugriffskennungen (API Keys) mehr dafür aus. Auch Amazon.com bietet seit 2002 auf seinen Portalseiten einen Webservice für verschiedene internetbezogene Dienstleistungen an, die Amazon Web Services.

Ein weiteres Beispiel ist die Interaktion zwischen Fluggesellschaften und Reisebüros. Die Fluggesellschaften stellen Möglichkeiten zum Nachschlagen bzw. Buchen von Flügen über einen Webservice bereit. Die Reisebüros bieten auf ihrer Webpräsenz Flüge verschiedener Fluggesellschaften an, von denen die Reisebüros zur Laufzeit über UDDI erfahren. Der Kunde kann auf der Webpräsenz des Reisebüros nun zentral Preise und Termine verschiedener Flüge vergleichen und direkt buchen.

Vor- Nachteile

Vorteile

- Die verwendeten offenen Standards vermeiden einige Lizenzkosten. Da zu diesen Standards auch die allgegenwärtigen internetbasierten Technologien gehören, lassen sie sich auch vielerorts einsetzen. Auch hier liegt ein Kostenvorteil.
- Webservices können faktisch auf jedes Übertragungsprotokoll aufsetzen. Bei einer hohen Anzahl von verschiedenen Nutzern im Internet wird üblicherweise HTTP zur Datenübertragung verwendet, da nur selten Probleme mit Firewalls auftreten. Dies ist ein Vorteil gegenüber vergleichbaren Technologien wie CORBA, DCOM oder auch Java RMI. Webservices sind wie beschrieben nicht an HTTP gebunden und lassen sich auch mit anderen Protokollen wie SMTP - zum Beispiel für asynchrone Übertragung - oder FTP - zum Beispiel bei sehr großen Nachrichten - übertragen und sind somit offen für verschiedene Anwendungsszenarien geeignet.
- Durch die Verwendung von bereits bestehenden und weit verbreiteten Internet-Standards (HTTP, XML etc.) entsteht eine offene und flexible Architektur, die unabhängig von den verwendeten Plattformen, Programmiersprachen und Protokollen ist. So können beispielsweise Windows-C#-Clients hinter einer Firewall mit Java-Servern, die auf Linux implementiert sind, kommunizieren. Die weit verbreiteten Standard-Protokolle ermöglichen eine Interoperabilität über jegliche Heterogenitäten im Internet hinweg.
- Die Barrieren zum Einstieg sind vergleichsweise niedrig.

Nachteile

- Die Hauptschwierigkeiten bei der Umsetzung von Webservices dürften Sicherheitsaspekte betreffen. So ist beim Transport zu beachten, dass wichtige Webservices verschlüsselt werden oder eine Authentifizierung stattfinden kann. Ob hier HTTPS ausreichend ist oder Lösungen wie XML Signature, XML-Encryption oder SAML zu bevorzugen sind, sollte abgewogen werden.
- Ein besonderes Augenmerk liegt auf der Performance. Diese wird durch XML, Parsen und Dateigröße negativ beeinflusst. Der Verwaltungsaufwand nimmt bei stark verteilten Systemen zu. Der Overhead ist teilweise erheblich.
- Es ist mehr Know-How erforderlich als z. B. mit Remote Procedure Call (RPC). Programmiersprachen, mit denen man Webservices einbinden will, brauchen spezielle Bibliotheken (z. B. DOM). Schnittstellen müssen genau definiert werden.

Anwendungsgebiete

Webservices stellen neue Ansätze im Rahmen von Enterprise Application Integration (EAI) und Grid-Computing dar. Das geplante Haupteinsatzgebiet liegt im Business-to-Business-Bereich (B2B). Geschäftsprozesse sollen problemlos über Unternehmensgrenzen hinweg abgewickelt werden. Eine Sprache hierfür ist Business Process Execution Language (BPEL), die es erlaubt zu orchestrieren.

Anwendungsmöglichkeiten

Web Services sind eine Technologie, die unter verschiedenen Aspekten verwendet werden kann. Hier die drei am weitesten verbreiteten:

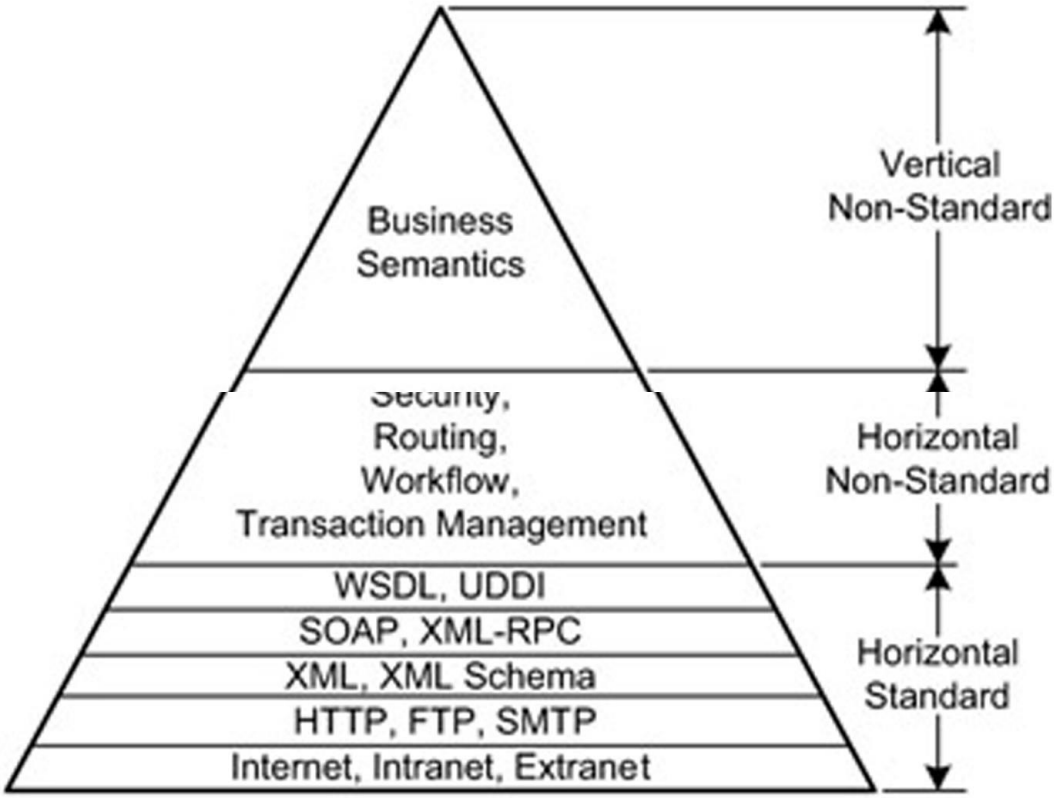
- RPC − Entfernter Funktionsaufruf: WSDL-basierend, die lose Kopplung ist nicht unbedingt gewährleistet
- SOAP − Eine Art Nachrichtendienst, WSDL als Kommunikationsinterface
- REST - Zustandsrepräsentationsübertragung − Der Versuch, das Interface auf eine Menge "definierter" Standard-Operationen (an HTTP angelehnt: GET, PUT, POST, DELETE) zu beschränken. Der Schwerpunkt liegt auf der Interaktion von zustandsbehafteten Ressourcen.

Implementierung über WSDL → SOAP HTTP oder direkt auf SOAP basierend

Standards

Die in folgender Abbildung gezeigte Web Service Pyramide umfasst die einzelnen Standards2, die für einen XML Web Service von Bedeutung sind. Ebenso stellt sie den hierarchischen Aufbau dieser Standards und den Grad der bisherigen Standardisierung dar.

Die ersten drei Ebenen bilden die Grundlage für die folgenden Sprachstandards eines Web Services. Das Internet, Intranet und Extranet stellt das Netzwerk dar, über das die Web Services anhand von XML basierten und durch XML Schemata1 definierte Nachrichten und Protokolle kommunizieren.



Kommunikationsprotokoll

Als Kommunikationsprotokoll für Web Services wird SOAP2 (**S**imple **O**bject **A**ccess **P**rotocoll) verwendet. Dieses Protokoll wurde vom W3C im Jahr 2000 in die Liste seiner Standards aufgenommen. Dabei gilt es, besonders die Unabhängigkeit von SOAP bzgl. des Transportprotokolls und die Erweiterbarkeit von SOAP aufgrund der Verwendung von XML als Sprachstandard hervorzuheben. Eine SOAP Nachricht besteht aus drei Komponenten:

- einem SOAP Envelope,
- optionalen SOAP Headern
- einem SOAP Body.

Der Body beinhaltet die eigentlich zu übermittelnde Nachricht, die Header-Elemente bieten zusätzliche Informationen bzgl. dieser Nachricht und der Envelope bildet den umschließenden Rahmen für das Body- und die Header-Elemente.

Schnittstellen Beschreibung

WSDL ist das Akronym für Web Service Definition Language. Es handelt sich dabei, wie

bei SOAP, um einen durch das W3C veröffentlichten XML-basierten Standard.

zur Beschreibung der Schnittstellen eines XML Web Services.

In Form eines WSDL-Dokuments ist es einem Web Service möglich auf standardisierte Weise seine Operationen, die dafür eingesetzten Datentypen und die Adresse unter der der Web Services zu finden ist, zu beschreiben und für potentielle Nutzer offen zu legen.

Standardisierte Dienstverzeichnis (UDDI)

Das Universal Description, Discovery, and Integration Protokoll (UDDI)5 beschreibt ein standardisiertes Verzeichnis, dass das Veröffentlichen und Suchen von Web Services ermöglicht. Über diverse Suchkriterien kann ein Dienst programmatisch gefunden und ausgewählt werden. Der „Anfrager“ erhält darauf das WSDL-Dokument des Web Services ausgehändigt.

UDDI Verzeichnisse treten in folgenden Variationen auf:

- öffentlich, global – für jedermann weltweit zugänglich
- privat – Zugang für eine geschlossene Benutzergruppe (auch unternehmensübergreifend)
- intern – Zugang nur innerhalb einer lokalen Domäne möglich

Sicherheitsaspekte

Da es sich bei XML und damit auch bei SOAP, um eine Darstellung von Daten in Textform

handelt und bei der Transportschicht dieser Nachrichten um offene Web-Standards wie http und TCP, sind sensible Daten, z.B. Passwörter, von Haus aus nicht vor dem Zugriff durch Dritte geschützt. Das Abfangen und Ausspionieren von Datenpaketen, die über TCP/IP oder HTTP transportiert werden, stellt dank geeigneter Werkzeuge selbst für nicht versierte Personen kein Problem dar.

Für die Lösung dieses Sicherheitsproblem gibt es diverse Ansätze:

- Sicherheit auf Transportebene unter der Verwendung von Secure-Socket-Layer (SSL) Dabei muss beachtet werden, dass es sich bei der Web Service Kommunikation um eine Ende-zu-Ende Verbindung handelt, d.h. zwischen dem Service, der eine Anfrage entgegen nimmt und dem Service, der die Anfrage bearbeitet, können mehrere Services liegen, die die Anfrage weiterleiten. Eine Absicherung durch SSL wäre dagegen nur Punkt-zu-Punkt basiert, also auf dem Transportweg zwischen den Services, gegeben.

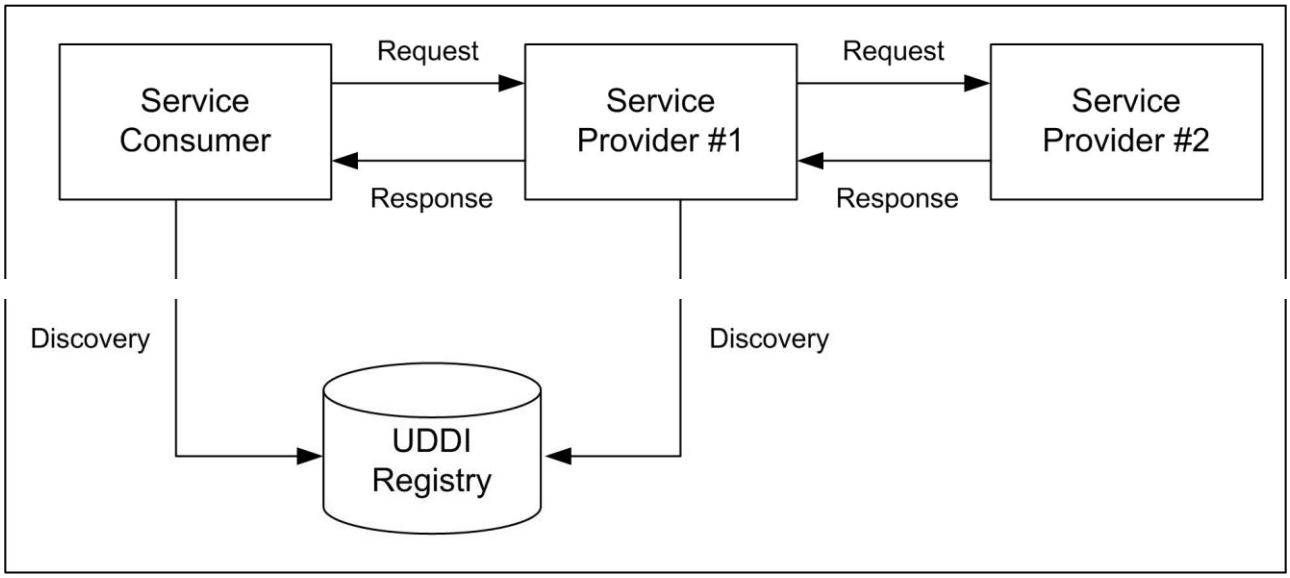
- Sicherheit auf der Datenebene (Chiffrierung der XML Nachricht mit Hilfe eines Verschlüsselungsalgorithmus)

- Einbinden des WS-Security Standards Der WS-Security Standard bietet über die Erweiterung von SOAP Header-Elementen Sicherheitsmaßnahmen an, die innerhalb der SOAP Nachricht geführt werden, und deshalb auch bei einer Weiterleitung über intermediary services erhalten bleiben.

Basismodell einer SOAP Architektur

Das Basismodell unterscheidet drei Rollen, die ein Dienst einnehmen kann.

- Service-Konsumenten (Service consumers)
- Service-Anbieter (Service providers)
- Service-Verzeichnisse (Service directories)



Service-Konsumenten

Ein Service-Konsument erfragt bei einem Service-Verzeichnis in diesem Falle ein UDDI Verzeichnis nach einer speziellen Dienstleistung. Vom Verzeichnis erhält der Konsument

die Schnittstellenbeschreibung (WSDL-Dokument) eines passenden Service-Anbieters ausgehändigt. Darauf basierend stellt der Konsument seine Anfrage an den Anbieter.

Service-Anbieter

Der Service-Anbieter registriert sich in einem Dienste Verzeichnis (z.B. bei einem UDDI Verzeichnis). Für die Registrierung wird eine Beschreibung seiner angebotenen Dienstleistung, sowie die die Schnittstellenbeschreibung (WSDL-Dokument) angegeben. Wie in der Abbildung gezeigt, kann ein Anbieter selbst zu einem Konsumenten werden.

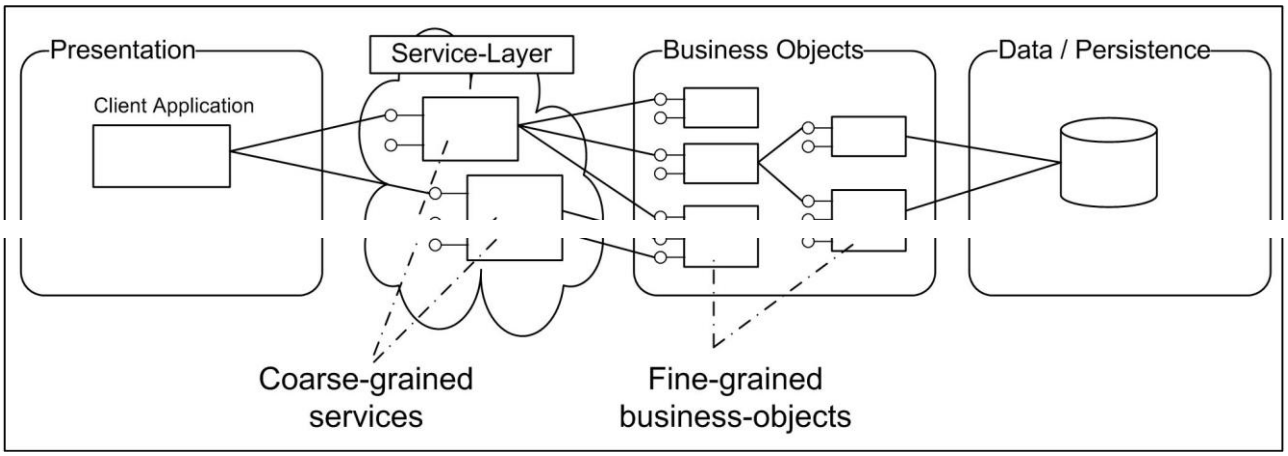
Service-Verzeichnisse

Ein Service-Verzeichnis stellt über eine standardisierte Schnittstelle eine Suche nach Dienstleistungen zur Verfügung. Damit die Dienstleistungen eines Anbieters im Verzeichnis erscheinen, muss dieser sich zuerst registrieren. Wurde eine passende Dienstleistung gefunden, übergibt das Verzeichnis die Schnittstellenbeschreibung des Service-Anbieters an den anfragenden Dienst (Service-Consumer).

Dieses Modell beschreibt auf einem abstrakten Level die Basisrollen, die Web Services innerhalb einer SOA einnehmen können. Darüber hinaus gibt es in einer komplexeren SOA diverse spezifischere Rollen, die durch Web Services repräsentiert werden.

Der Service-Layer

Der Service-Layer wird als neue logische Schicht in das bisherige Schichtenmodell einer Applikationsarchitektur eingezogen. Diese Schicht befindet sich zwischen der Präsentationsschicht und der Anwendungslogik und umschließt alle Web Services einer Applikation bzw. eines Unternehmens. Der Service-Layer wird in der Abbildung als Wolke dargestellt. Dadurch soll die Ortstransparenz von Web Services verdeutlicht werden.



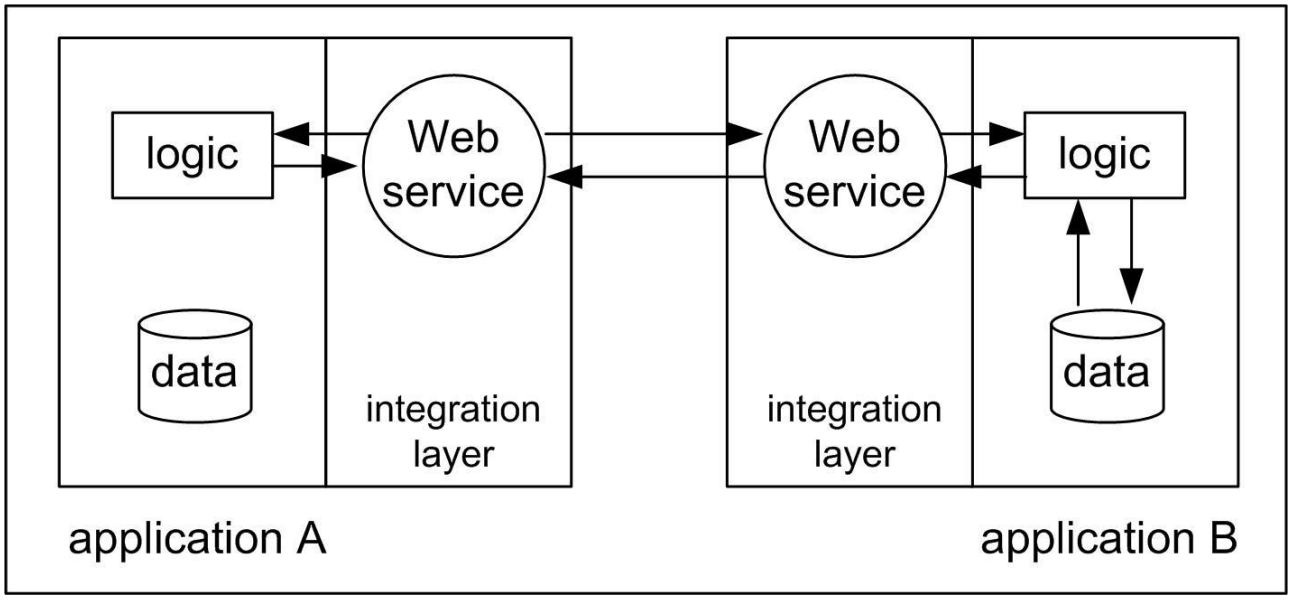
Zu den Aufgaben des Service-Layers zählen:

- Standardisierter zentraler Anlaufpunkt für Anfragen an einen Service
- Kapselung, Komposition und Indirektion der zu Grunde liegenden Dienste
- Je nach Menge der zu Grunde liegenden Dienste sollte der Service-Layer ein Dienste Verzeichnis (z.B. UDDI) führen

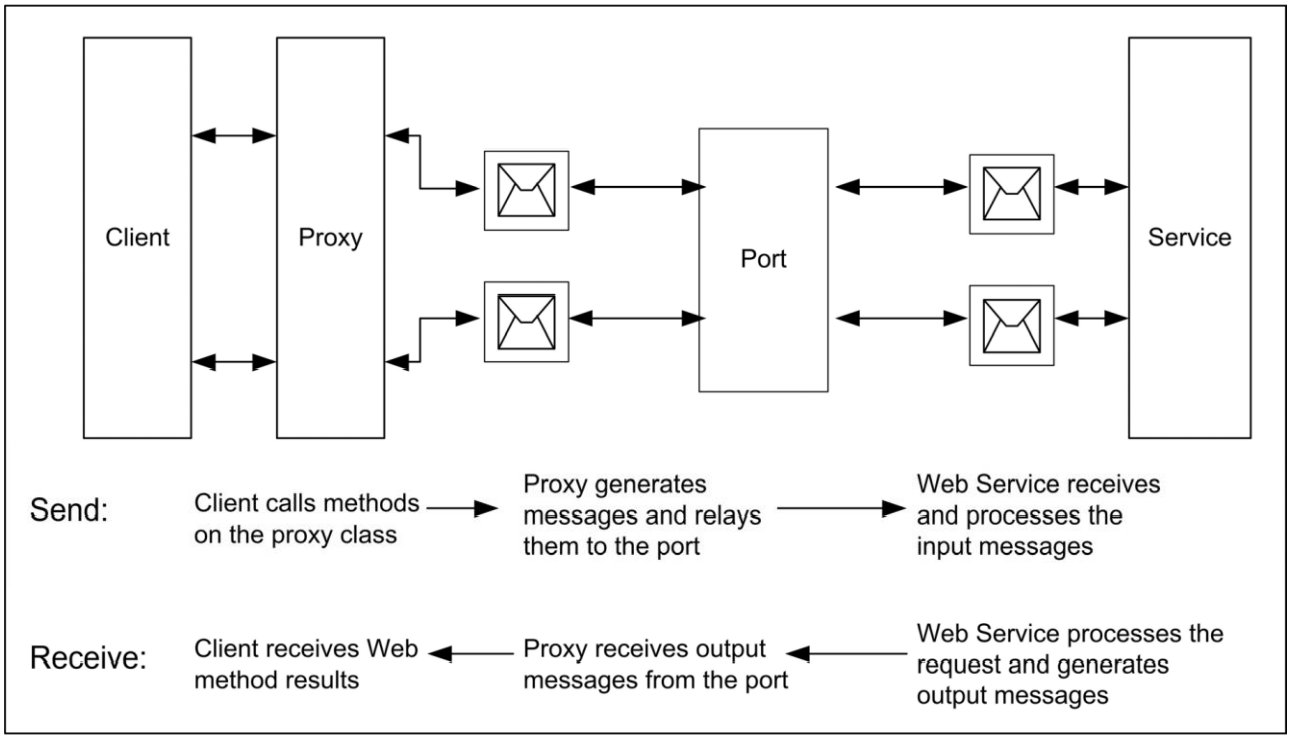
Innerhalb eines Service-Layers gibt es zwei Hauptrollen – Service Fassade und Service Agent, die von Web Services eingenommen werden können. Diese beiden Rollen sollen nun näher beschrieben werden.

Service-Adapter für heterogene Systeme

Diese Aufbereitung erledigt ein so genannter Service-Adapter. Dieser Adapter wird direkt auf die Applikationslogik-Schicht aufgesetzt und konvertiert anhand von Serialisierung und Deserialisierung eingehende Anfragen in das jeweilige spezifische Format des zu Grunde liegenden Systems.

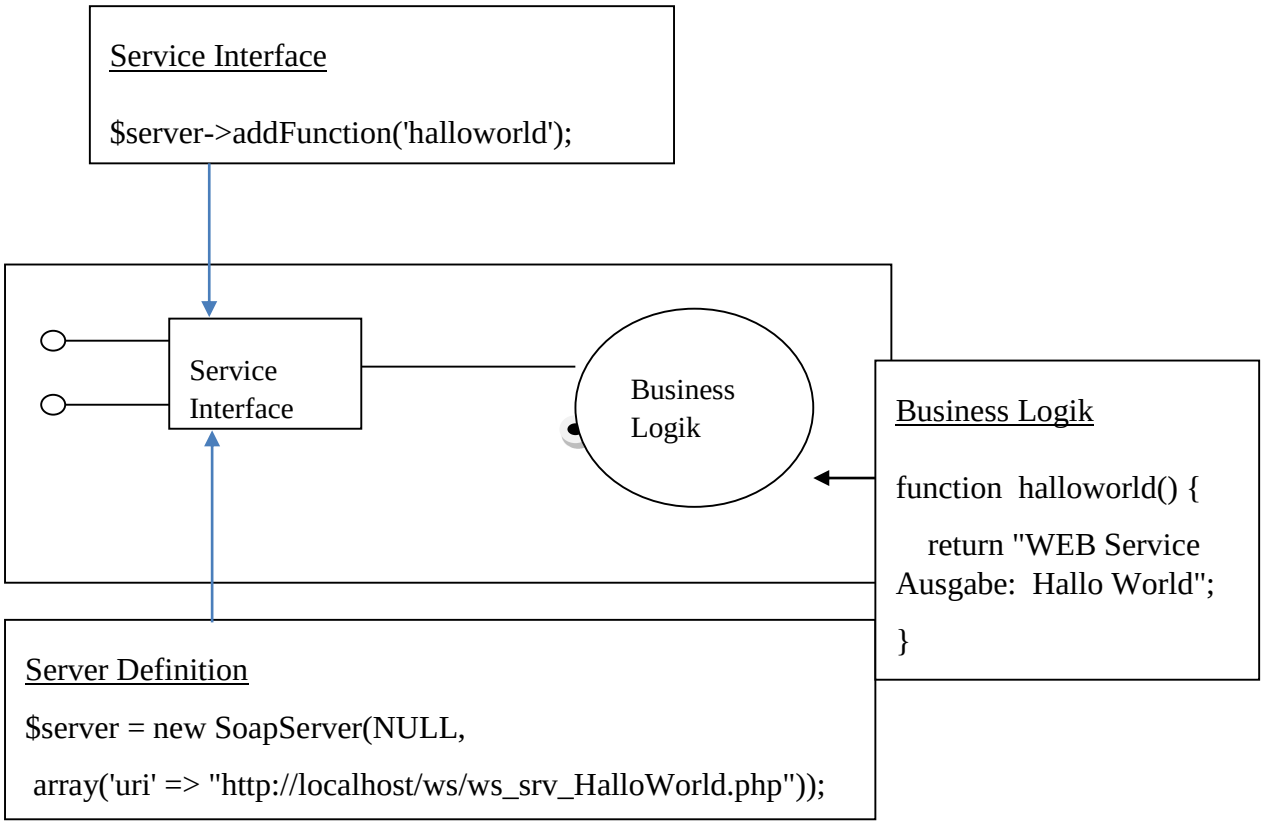


Anwendungsbeispiel Service- orientierter Architekturen



Einfaches Beispiel einer Web Service Implementation in PHP (Ohne WDSL)

Service Provider (Server)



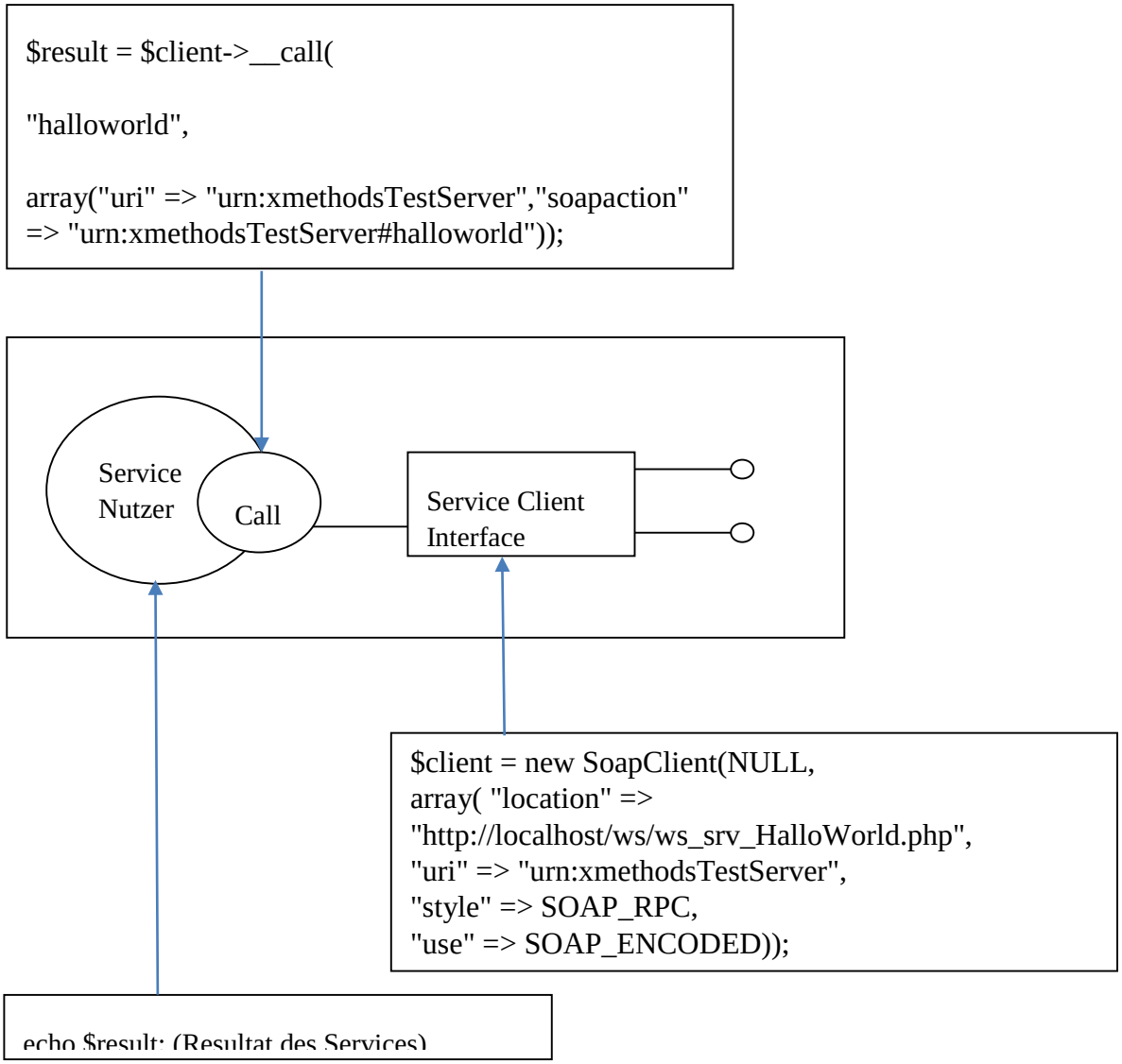
PHP Code für den WEB Service Server Funktionen

Service Beschreibung:

Der Service generiert an den Client einen Kommentar mit dem Inhalt „Hallo World“.

```
<?php
  $server = new SoapServer(NULL,
     array('uri' => "http://localhost/ws/ws_srv_HalloWorld.php"));
  $server->addFunction('halloworld');        //Funktion zum Server hinzufügen
  $server->handle();                         //Hier wird die Abfrage abgearbeitet
  function  halloworld() {
      return "WEB Service Ausgabe:  Hallo World";
  }
?>
```

Web Service Benutzer (Client)



WEB Service Client Beispiel

```
<?php
$client = new SoapClient(NULL,
array(
"location" => "http://localhost/ws/ws_srv_HalloWorld.php",
"uri" => "urn:xmethodsTestServer",
"style" => SOAP_RPC,
"use" => SOAP_ENCODED));
```

Parameter Übergabe

a) WEB Service Client Parameter Definition

```
$client = new SoapClient(NULL,
  array(
  "location" => "http://localhost/ws/ws_srv_math.php",
  "uri" => "urn:xmethodsTestServer",
  "style" => SOAP_RPC,
  "use" => SOAP_ENCODED
));

$parameters = array(
new SoapParam('VariableInhalt', 'Variable1'),
new SoapParam('VariableInhalt', 'Variable1'));

$result = $client->__call(
"addiere",
$parameters,
array(
"uri" => "urn:xmethodsTestServer",
"soapaction" => "urn:xmethodsTestServer#addiere"     //irgendein Platzhalter
));
```

Beispiel

```
  $client = new SoapClient(NULL,
    array(
    "location" => "http://localhost/ws/ws_srv_math.php",
    "uri" => "urn:xmethodsTestServer",
    "style" => SOAP_RPC,
```

```
  "use" => SOAP_ENCODED
));

$parameters = array(
new SoapParam('50', 'summand1'),  50 wird in der Variable summand1 übergeben
new SoapParam('20', 'summand2')); 20 wird in der Variable summand1 übergeben

$result = $client->__call(
"addiere",
$parameters,
array(
"uri" => "urn:xmethodsTestServer",
"soapaction" => "urn:xmethodsTestServer#addiere"     //irgendein Platzhalter
));
```

b) WEB Service Server Parameter Definition

```
$server = new SoapServer(NULL,
 array('uri' => "http://localhost/ws/ws_srv_math.php"));
$server->addFunction('addiere');    //Funktion zum Server hinzufügen
$server->handle();                  //Hier wird die Abfrage abgearbeitet


function  addiere($summand1, $summand2) {
    return $summand1 + $summand2;
}
```

c) WEB Services mit Stored Procedure (Client)

```
<?php
$server = "localhost";
$user = "root";
$pw = "";

$client = new SoapClient(NULL,
array(
'location' => "http://localhost/ws/ws_srv_db_storedprocedure.php",
'uri' => "urn:xmethodsTestServer",
'style' => SOAP_RPC,
'use' => SOAP_ENCODED
));

$parameters = array(new SoapParam($server, 'server'), new SoapParam($user, 'us-
er'), new SoapParam($pw, 'pw'));

// $result = $client->__call("ws_get_mitarbeiter", $parameters, array('uri' =>
"urn:xmethodsTestServer", 'soapaction' =>
"urn:xmethodsTestServer#ws_get_mitarbeiter"));


// kurze version
$result = $client->ws_get_mitarbeiter($server,$user,$pw);

print_r($result);
?>
```

d) WEB Services mit Stored Procedure (Server)

```
<?php
$server = new SoapServer(NULL, array (
'uri' => "http://localhost/ws/ws_srv_db_storedprocedure.php"
```

```
));
$server->addFunction('ws_get_mitarbeiter');
$server->handle();

function ws_get_mitarbeiter($server, $user, $pw) {
mysql_connect($server, $user, $pw,0,65536);
mysql_select_db("mp151");
// $result = mysql_query("SELECT name, vorname, tel, email, gehalt, stellung FROM
tbl_mitarbeiter");
$result = mysql_query("call sps_get_mitarbeiter()");

while ($row = mysql_fetch_array($result)) {
$mitarbeiter[] = $row;
}
return $mitarbeiter;
}
?>
```

Abkürzungen

ASP	**A**ctive **S**erver **P**ages
B2B	**B**usiness **to B**usiness
CAS	**C**entral **A**uthentication **S**ervice
COM	**C**omponent **O**bject **M**odel
CORBA	**C**ommon **O**bject **R**equest **B**roker **A**rchitekture
DCOM	**D**istributed **C**omponent **O**bject **M**odel
ESA	**E**nterprise **S**ervices **A**rchitecture
HTML	**H**yper**t**ext **M**arkup **L**anguage
HTTP	**H**yper**T**ext **T**ransfer **P**rotocol
HTTPS	**H**yper**T**ext **T**ransfer **P**rotocol **s**ecure
JSP	**J**ava**S**erver **P**ages
LDAP	**L**ightweight **D**irectory **A**ccess **P**rotocol
OASIS	**O**rganization for the **A**dvancement of **S**tructured **I**nformation **S**tandards
ODBC	**O**pen **D**ata**B**ase **C**onnectivity
PHP	**H**y**p**ertext **P**reprocessor
RDF	**R**esource **D**escription **F**ramework
RFC	**R**equest **F**or **C**omments
RMI	**R**emote **M**ethod **I**nvocation
RPC	**R**emote **P**rocedure **C**all
RSS	**RDF S**ite **S**ummary
SMTP	**S**imple **M**ail **T**ransfer **P**rotocol
SOA	**S**ervice-**o**rientierte **A**rchitektur
SOAP	**S**imple **O**bject **A**ccess **P**rotocol
SQL	**S**tructured **Q**uery **L**anguage
SSL	**S**ecure **S**ocket **L**ayer
SSO	**S**ingle **S**ign-**O**n
TCP/IP	**T**ransmission **C**ontrol **P**rotocol / **I**nternet **P**rotocol
UDDI	**U**niversal **D**escription, **D**iscovery and **I**ntegration
W3C	**W**orld **W**ide **W**eb **C**onsortium
WS	**W**eb **S**ervice

WS-I **W**eb **S**ervices **I**nteroperability Organization

WSDL **W**eb **S**ervices **D**efinition **L**anguage

XML e**X**tensible **M**arkup **L**anguage

XSD **X**ML **S**chema **D**efinition

XSL e**X**tensible **S**tylesheet **L**anguage

XSLT **XSL** for **T**ransformation