Prof. DI Dr. Erich Gams

Web Services

Einführung, Aufbau, Kleines Beispiel

Informationssysteme - htl-wels

Übersicht ☛ Was lernen wir?

- Motivation
- Einsatzgebiete
- SOAP Web Services-Aufbau
- Kommunikation
- Kleines Beispiel
- REST Web Services-Aufbau
- Kommunikation
- Kleines Beispiel

Einführung & Motivation

- Ansprechen entfernter Ressourcen oder Aufrufen entfernter Methoden
- Standards: RMI, CORBA, DCOM, RPC
  - Erfordern bestimmte offene Ports
  - Oft ist der einzige offene Port 80 (für HTTP)
- Anforderungen an Anwendungstypus
  - Port 80 Kommunikation
  - Lösungen sollten plattformübergreifend funktionieren.
  - Zugriff ohne aufwändige Generatoren zur Testvereinfachung

Einführung Web Services

- Dienst über‘s Internet bereitgestellt - „Software as a service“
  - durch standardbasierte Protokolle wie HTTP nutzbar
  - durch Uniform Resource Identifier (URI) eindeutig identifizierbar
- Beschreibung, Verzeichnisdienste und Nachrichtenaustausch XML-basiert (ähnlich XML-RPC)
- Client plattformunabhängig
- als Middleware im Bereich E-Business von zunehmender Bedeutung
- bekannte Beispiele: Web Services von google, amazon, ebay, ….

Web Services have Two Types of Uses

- **Reusable application-components.**
  - There are things applications need very often. So why make these over and over again?
  - Web services can offer application-components like: currency conversion, weather reports, or even language translation as services.
- **Connect existing software.**
  - Web services can help to solve the interoperability problem by giving different applications a way to link their data.
  - With Web services you can exchange data between different applications and different platforms

Vorteile der Web-Services-Technologie

- Einfache und flexible Gestaltung von Geschäftsprozessen
- Modellierung und Wiederverwendung funktionaler Einheiten
- Web-Services-Technologie besonders für die technische Realisierung von serviceorientierten Architekturen geeignet
- Integration existierender Software durch Einziehen einer Web-Services-Schicht.
  - Aufsetzen von Web Services als so genannte "Wrapper" auf existierende Legacy-Software.
  - Eingespielte Geschäftsprozesse und bereits implementierte Funktionalität kann weiterverwendet und direkt im Kontext von Intranet, Extranet und Internet genutzt werden.

SOAP Web Services

- SOAP ist ein Netzwerkprotokoll zum Austausch von XML-basierten Nachrichten über ein Computernetzwerk.
- Es kodiert die Nachricht in XML und überträgt Aufrufe über standardisierte Internet-Technologien an die Methode des Webservices sowie das Resultat der Anfrage.
- Als Transportmedium wird meistens HTTP verwendet.
- Durch viele Erweiterungen ist SOAP heute sehr mächtig und komplex.
- SOAP stand früher für *Simple Object Access Protocol*. Inzwischen wird SOAP nicht mehr als Akronym, sondern als Eigenname verwendet.

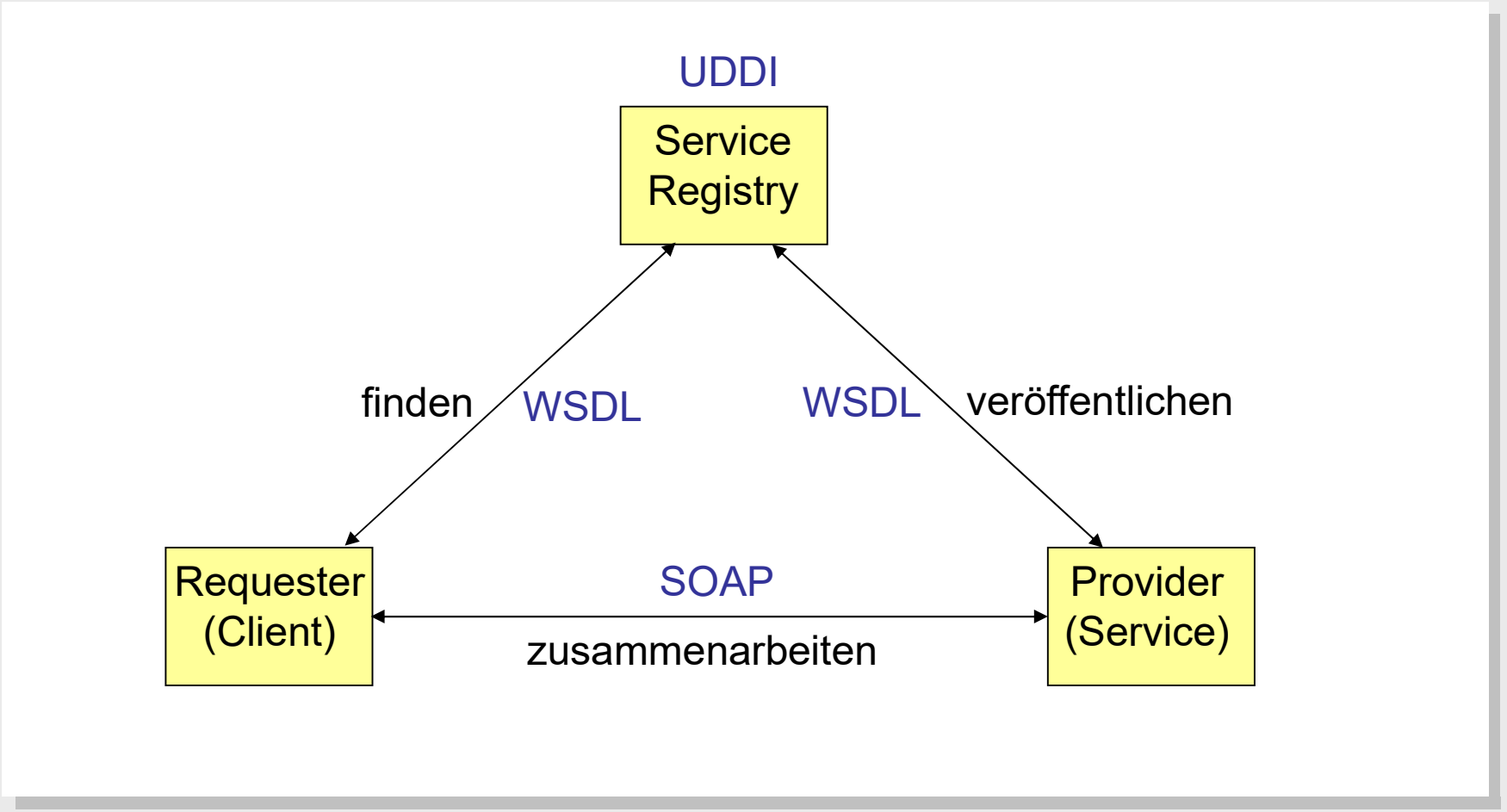
SOAP Grundidee

- Die Grundidee von SOAP-Webservices ist, einen Request in Form einer XML-Datenstruktur an einen definierten Server zu schicken, wobei der Server einen Router enthält, der die XMLDatenstruktur interpretiert und die aufzurufende Operation am Server erkennt.
- Die benötigten Parameter für die Operation werden aus den XML-Daten ausgelesen und dem Operationsaufruf mitgegeben.
- **SOAP, WSDL und UDDI** bilden die drei Kern-Standards für SOAP Webservices

Aufbau

- **SOAP**
  - **Simple Object Access Protocol** (bis SOAP1.2)
  - Protokollstandard des W3C zur **Kommunikation**
  - **beschreibt Art und Weise, wie Inhalte und Daten übertragen werden**
- **WSDL**
  - **Web Service Description Language**
  - Sprache zur Beschreibung der unterstützten Methoden und **Parameter** (öffentliche Schnittstellenbeschreibung)
- **UDDI**
  - **Universal Description, Discovery, and Integration**
  - **Verzeichnisdienst** zur Registrierung/Veröffentlichung von Web Services
  - ermöglicht dynamisches Finden von Web Service

Kommunikation



Vor- und Nachteile

- betriebssystem- und plattformunabhängige Schnittstelle
- Sehr große XML-Dokumente
- Nachteil für Umgebungen in der performante Umgebung gefordert wird
- Client kann sich mit einem Server verbinden und Aufruf starten, obwohl die Berechtigung fehlt
- Im Klartext übertragene Nachrichten

Aufbau einer SOAP-Nachricht

- SOAP-Messages enthalten einen **Envelope**, einen **Header** und einen **Body**
- **SOAP Envelope:**
  - umfasst den Header und den Body.
- **SOAP Header:**
  - Optionales Element
  - enthält Informationen zur Verarbeitung der Nachricht
    - z.B. Informationen über das Routing der Nachricht
- **SOAP Body**
  - enthält die Informationen, welche an den Empfänger gesendet werden sollen (Parameter der Request- / Reply-Nachricht).
  - Ein SOAP-Body muss als wohlgeformtes XML-Dokument vorliegen

Beispiel

HTTP-Request inkl. SOAP Nachricht

Web Service Description Language (WSDL)

- Die Web Service Description Language (WSDL) ist ein XML-Derivat zur Beschreibung der Schnittstelle eines Webservices
- definiert die Nachrichtenstrom-Formate und Funktionsaufrufe.
- Das WSDL-Dokument beschreibt also im Wesentlichen, welche Methoden der Webservice anbietet, mit welchen Parametern sie aufzurufen sind, was sie zurückliefern und wie Kontakt zu meinem Webservices aufgenommen werden kann.

Universal Description, Discovery and Integration (UDDI)

- Um einen Service benutzen zu können, muss die Anwendung dem Serviceanbieter bekannt sein. Dies wird durch UDDI ermöglicht.
- 3 Akteure
  - Der Service Consumer möchte ein Service benutzen.
  - Der Service Provider bietet ein Service an.
  - Das Service Discovery (UDDI), ein Verzeichnis, ist der Vermittler
- UDDI
  - weltweite Registrierungsstelle von Webservices
  - bezeichnet einen standardisierten Verzeichnisdienst (Veröffentlichung und Entdeckung) von Adress- und Produktdaten
  - Bietet Anwendungs-Schnittstellen der verschiedenen Webservices-Anbieter.
- Der Verzeichnisdienst besitzt eine SOAP-Schnittstelle. Er enthält Unternehmen, ihre Daten und ihre Services.

Web services @ work

- Verschiedene Frameworks
  - Apache Axis 2
  - Apache CXF
  - Spring-WS
  - Metro
  - usw…
- Oder…… Java 6 und neuer

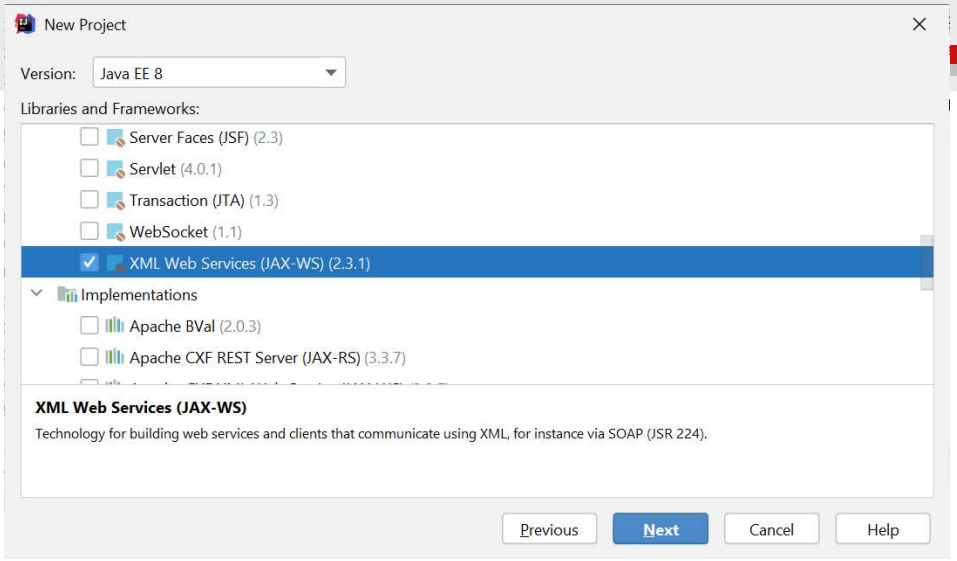
Web services mit Java 6

- Seit **Java 6 (JDK 6)** kann man schnell einen **Web Service** mit Java (**JAX-WS 2**) definieren.
- Jede Klasse kann ein Web Service werden.
- Java Annotations helfen eine Klasse als WebService zu deklarieren.

Annotationen

- **@WebService**
  - Jede Web Service Implementierung muss diese Klassen Annotation besitzen
- **@SOAPBinding**
  - Dokument oder RPC
- **@WebMethod**
  - Macht eine Methode zur Web Service Operation
- **@WebParam**
  - Beschreibt die Parameter genauer
- **@WebResult**
  - Bestimmt die Rückgabe einer Web Service Methode genauer
- **@OneWay**
  - Asynchroner Aufruf

Projekt in IntelliJ




```
<groupId>com.example</groupId>
<artifactId>INSY-SOAPWebServicesDemo</artifactId>
<version>1.0-SNAPSHOT</version>
<name>INSY-SOAPWebServicesDemo</name>
<packaging>jar</packaging>
```

Implementation der WebService Klasse

```
package webservice.service;

import javax.jws.WebService;
import javax.jws.soap.SOAPBinding;
import javax.jws.soap.SOAPBinding.Style;

@WebService (serviceName="CalculatorWS") // Web Service
@SOAPBinding(style=Style.RPC) // Aufruf mit Parametern, Rückgabewerte

public class Calculator {
  @WebMethod(operationName="Rechner")
  public long addValues(
                          @WebParam(name="Operand_1") int val1,
                        @WebParam(name="Operand_2" int val2)
{
    return val1 + val2;
  }
}
```

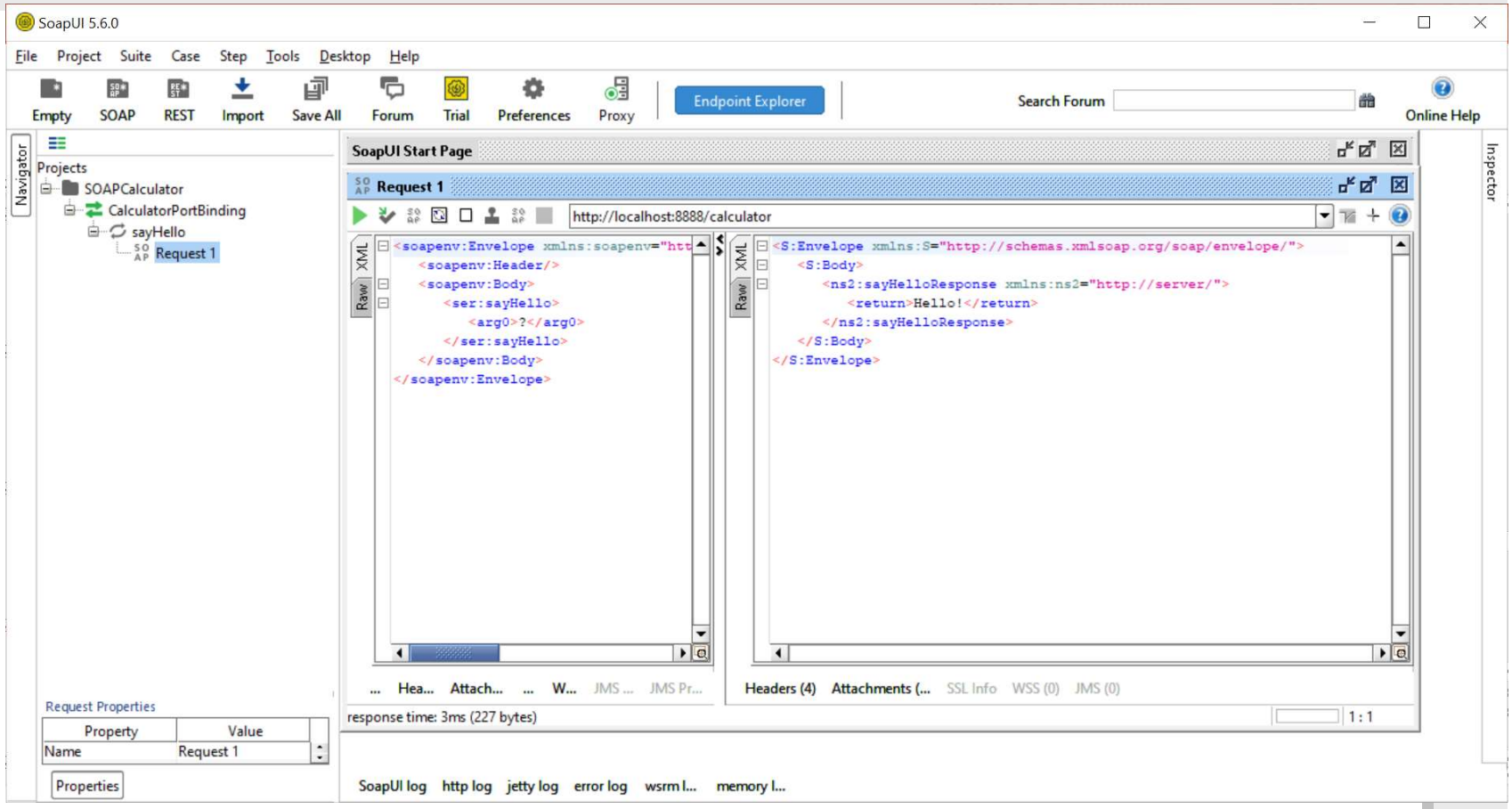
Implementation des WebService Servers

```
package webservice.server;

import javax.xml.ws.Endpoint;
import de.theserverside.webservice.service.Calculator;

public class CalculatorServer {
  public static void main (String args[]) {
    Calculator server = new Calculator();
    Endpoint endpoint =                                  // publizieren
      Endpoint.publish("http://localhost:8080/calculator", server);
  }
}
```

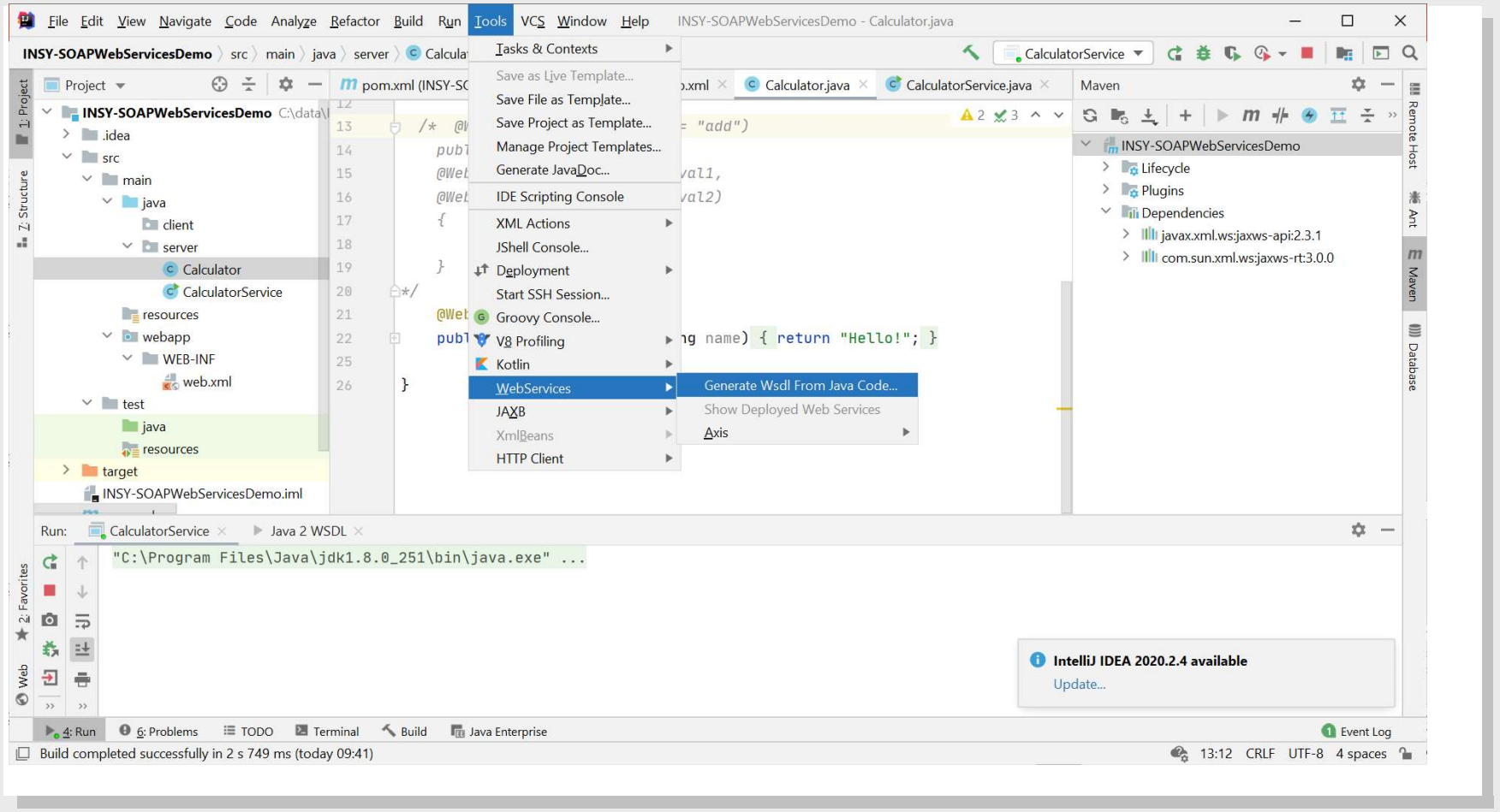
WSDL aufrufen

- http://localhost:8080/calculator?wsdl

SOAPUI


SoapUI 5.6.0
File Project Suite Case Step Tools Desktop Help
Empty SOAP REST Import Save All Forum Trial Preferences Proxy
Endpoint Explorer
Search Forum
Online Help
Navigator
Projects
SOAPCalculator
CalculatorPortBinding
sayHello
Request 1
SoapUI Start Page
Request 1
http://localhost:8888/calculator
<soapenv:Envelope xmlns:soapenv="htt
<soapenv:Header/>
<soapenv:Body>
<ser:sayHello>
<arg0>?</arg0>
</ser:sayHello>
</soapenv:Body>
</soapenv:Envelope>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
<S:Body>
<ns2:sayHelloResponse xmlns:ns2="http://server/">
<return>Hello!</return>
</ns2:sayHelloResponse>
</S:Body>
</S:Envelope>
Request Properties
Property
Value
Name
Request 1
Properties
response time: 3ms (227 bytes)
1 : 1
SoapUI log
http log
jetty log
error log
wsrm l...
memory l...
Inspector

IntelliJ und WSDL



Multithreading

```
ExecutorService es = Executors.newFixedThreadPool( nThreads: 5);
Endpoint ep2 = Endpoint.create(new Calculator());
ep2.setExecutor(es);
ep2.publish( address: "http://localhost:8080/test");
```

WSDL Beschreibung

- Aufruf:
  - http://localhost:8080/calculator?wsdl
- Ausgabe:

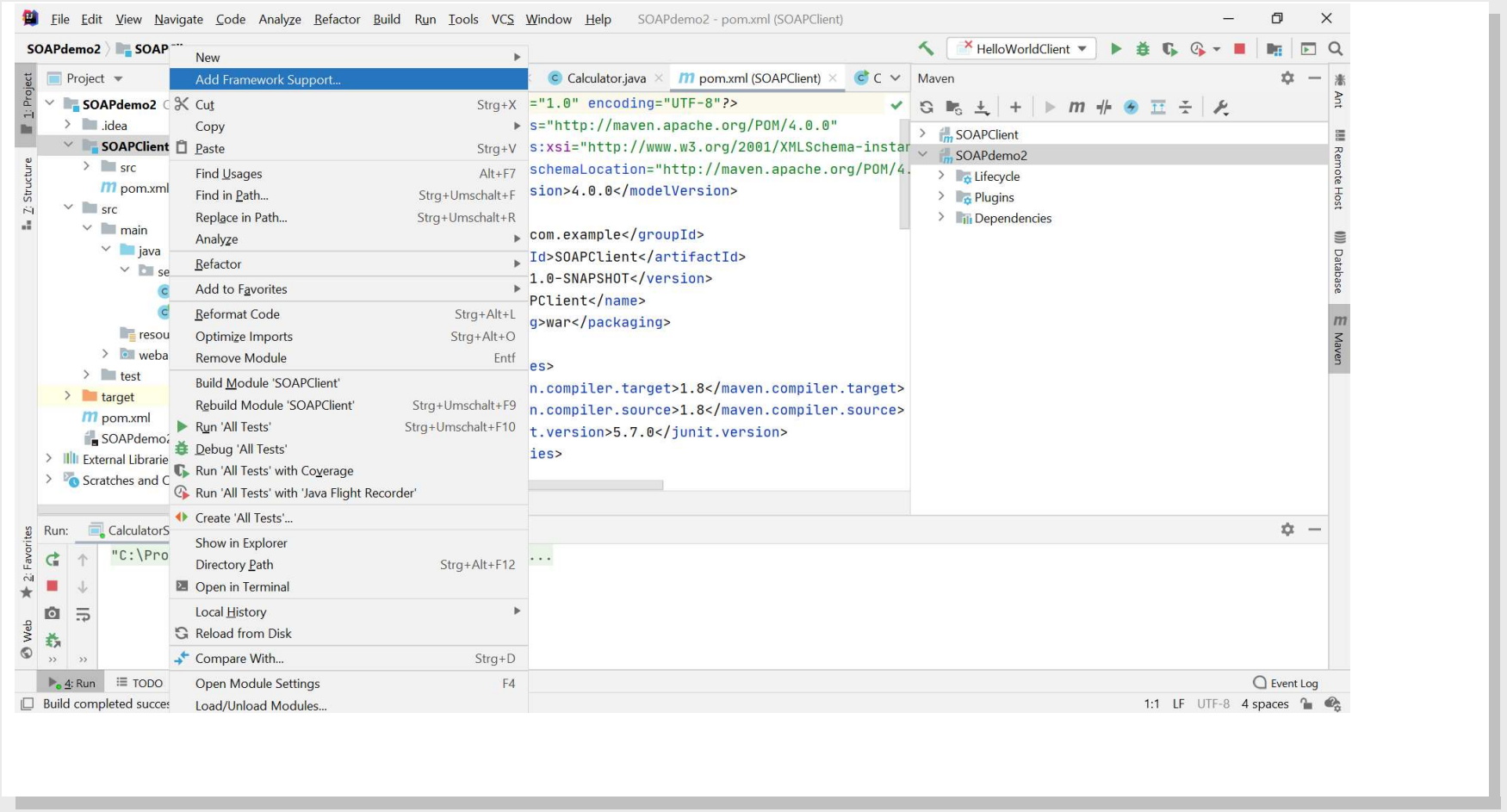
```
....
<types></types>
<message name="addValues">
 <part name="arg0" type="xsd:int"></part>
 <part name="arg1" type="xsd:int"></part>
</message>
<message name="addValuesResponse">
 <part name="return" type="xsd:long"></part>
</message>
<portType name="Calculator">
 <operation name="addValues" parameterOrder="arg0 arg1">
  <input message="tns:addValues"></input>
  <output message="tns:addValuesResponse"></output>
 </operation>
</portType>
.....
```

Implementation des WebService Clients

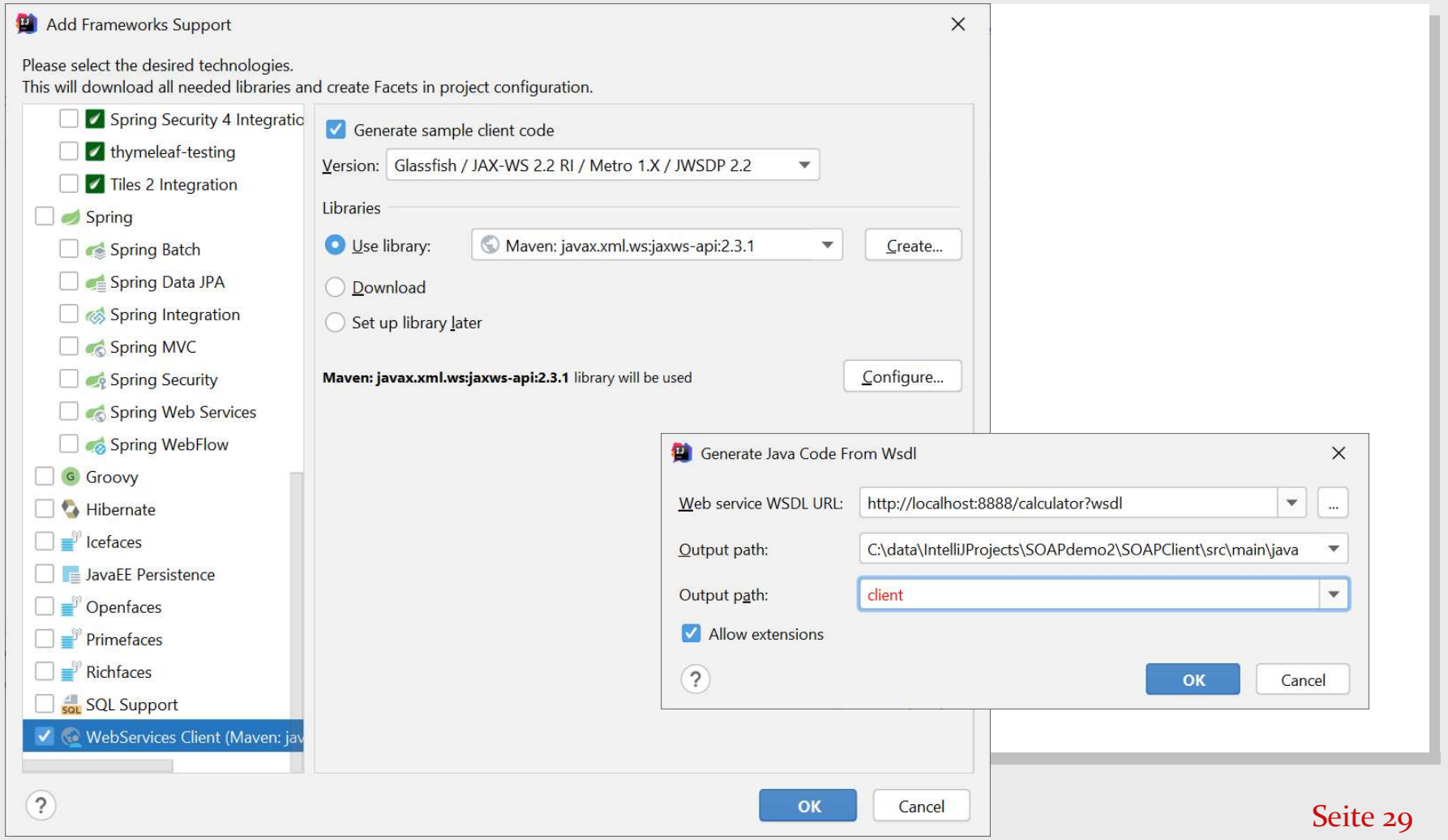
- wsimport -keep http://localhost:8080/calculator?wsdl
  - -d pfad
  - -p Paket
  - -keep Generierung

- Calculator.java // Interface
- Calculator.class
- CalculatorService.java // übernimmt Kommunikation zum Server
- CalculatorService.class

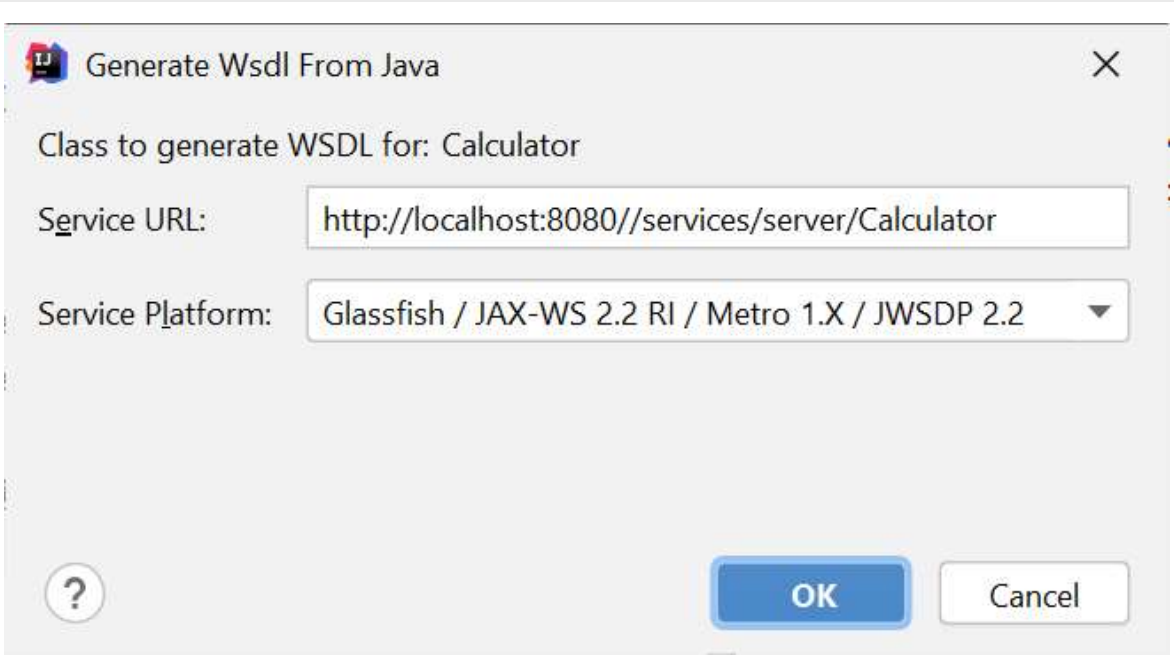
IntelliJ: Generate Client



Module: Add framework support



IntelliJ und WSDL



Implementation des WebService Clients

```
package webservice.client;

import webservice.service.Calculator;
import webservice.service.CalculatorService;

public class CalculatorClient {
   public static void main(String args[]) {
      CalculatorService service = new CalculatorService();
            //erzeugt Handle auf Service
      Calculator calculator = service.getCalculatorPort();
      System.out.println("Summe: " + calculator.addValues(17, 13));
   }
}
```

Aufgaben

- Beispiel: BMI Berechnung
  - Methodenname: body-mass-index
  - Resultatname: your-bmi
  - Parameternamen: height, weight
- Beispiel: Bodenfliesen
  - Für einen Bodenleger soll ein Programm geschrieben werden, das die Anzahl der benötigten Bodenfliesen für einen einzugebenden Raum berechnet.
  - Die Seitenlänge u. -breite der Bodenfliese ist zu übergeben. Es wird 10% Ausschuss dazugerechnet. Es soll die Anzahl der Bodenfliesen berechnet werden.
  - Methodenname:number-of-tiles