JUST DO IT
HÖHERE ABTEILUNG FÜR INFORMATIONSTECHNOLOGIE

tgm
Die Schule der Technik

DEZSYS-07

SOA and RESTful Webservices

Systemtechnik Unterricht
5BHITT 2015/16

Burkhard Hampl & Simon Wortha

Version 1.0

Note:

Betreuer: Micheler/Borko

Begonnen am 4. Dezember 2015

Beendet am 6. Jänner 2016

Inhaltsverzeichnis

1 Einführung

1.1 Ziele

Ziel ist es eine Wissensdatenbank zu erstellen, auf diese mit RESTful Wbservices über eine Weboberfläche zugegriffen werden kann. Weiters soll ein SOA Webservice in Java entwickelt werden, welcher für die Suche verantwortlich ist.

1.2 Voraussetzungen

Voraussetzung für diese Übung sind folgende Referate:

- RESTful Services und Json von Stefan Geyer
- SOA, SOAP und EAI von Niklas Hohenwarter

1.3 Aufgabenstellung

Das neu eröffnete Unternehmen iKnow Systems ist spezialisiert auf Knowledgemanagement und bietet seinen Kunden die Möglichkeiten Daten und Informationen jeglicher Art in eine Wissensbasis einzupflegen und anschließend in der zentralen Wissensbasis nach Informationen zu suchen (ähnlich wikipedia).

Folgendes ist im Rahmen der Aufgabenstellung verlangt:

- Entwerfen Sie ein Datenmodell, um die Eintraege der Wissensbasis zu speichern und um ein optimitiertes Suchen von Eintraegen zu gewaehrleisten. [2Pkt]
- Entwickeln Sie mittels RESTful Webservices eine Schnittstelle, um die Wissensbasis zu verwalten. Es muessen folgende Operationen angeboten werden:

  - Hinzufuegen eines neuen Eintrags

  - Aendern eines bestehenden Eintrags

  - Loeschen eines bestehenden Eintrags

- Alle Operationen muessen ein Ergebnis der Operation zurueckliefern. [3Pkt]
- Entwickeln Sie in Java ein SOA Webservice, dass die Funktionalitaet

Suchen anbietet und das SOAP Protokoll einbindet. Erzeugen Sie fuer dieses Webservice auch eine WSDL-Datei. [3Pkt]

- Entwerfen Sie eine Weboberflaeche, um die RESTful Webservices zu verwenden. [3Pkt]
- Implementieren Sie einen einfachen Client mit einem User Interface (auch Commandline UI moeglich), der das SOA Webservice aufruft. [2Pkt]
- Dokumentieren Sie im weiteren Verlauf den Datentransfer mit SOAP. [1Pkt]
- Protokoll ist erforderlich! [2Pkt]

Info:

Gruppengroesse: 2 Mitglieder

Punkte: 16

Zum Testen bereiten Sie eine Routine vor, um die Wissensbasis mit einer 1 Million Datensaetze zu fuellen. Die Datensaetze sollen mindestens eine Laenge beim Suchbegriff von 10 Zeichen und bei der Beschreibung von 100 Zeichen haben! Ist die Performance bei der Suche noch gegeben?

Links:

JEE Webservices:

http://docs.oracle.com/javaee/6/tutorial/doc/gijti.html

Apache Web Services Project:

http://ws.apache.org/

Apache Axis/Axis2:

http://axis.apache.org

http://axis.apache.org/axis2/java/core/

IBM Article: Java Web services - JAXB and JAX-WS in Axis2:

http://www.ibm.com/developerworks/java/library/j-jws8/index.html

2 Ergebnisse

2.1 Datenbank

Wir haben uns für eine PostgreSQL Datenbank entschieden. Dort muss eine neue Datenbank angelegt werden und ein User der Berechtigungen für diese hat. Die Information den Datenbank-Server (IP, User, Datenbank) müssen in der Datei **application.properties** festgelegt werden. Dieses File liegt der Abgabe aber NICHT bei. Dafür gibt es aber die Datei **apllication.properties.example** in welcher die Anpassungen gemacht werden können und anschließend muss die Datei dann als **application.properties** gespeichert werden.

Die 1 Millionen Test-Datensätze werden von einer einfachen for-Schliefe erzeugt.

2.2 REST-API

Die Rest API wurde mittels Spring Boot implementiert [2]. Als erstes wurde die Knowledge Entry Klasse erstellt, welche unsere Datenbasis darstellt. Dann wurde das Repository erstellt welches für den Zugriff auf die Datenbank benötigt wird. Nun konnten die benötigen Requests mit den Responses gemapped werden.

2.3 REST-Client

Der Client wurde mittels Javascript implementiert [3], um die statischen Dateien mittels thymeleaf auszuliefern, wurde thymeleaf, eine template engine, verwendet.

2.4 SOA-Webservice

Der SOA-Webservice wurde auch mittels Spring Boot implementiert [1]. Dazu wurde die WSDL erstellt und daraus dann die Klassen generiert. Nun konnten die Request mittels der Endpoint mit den Responses gemapped werden. Zur Konfiguration wurde noch eine WebServiceConfig erstellt.

2.5 SOA-Client

Zuerst muss eine Verbindung zum Service aufgebaut werden.

```
// Create SOAP Connection
SOAPConnectionFactory soapConnectionFactory =
SOAPConnectionFactory.newInstance();
SOAPConnection soapConnection = soapConnectionFactory.createConnection();
```

Nun kann eine Message erstellt werden.

```
MessageFactory messageFactory = MessageFactory.newInstance();
SOAPMessage soapMessage = messageFactory.createMessage();
SOAPPart soapPart = soapMessage.getSOAPPart();
```

Hier ist zu beachten, dass eine SOAP Message einen Header, einen Body und ein Envelope benötigt.

```
// SOAP Envelope
SOAPEnvelope envelope = soapPart.getEnvelope();
envelope.addNamespaceDeclaration("knowledge", NAMESPACE);


// SOAP Body
SOAPBody soapBody = envelope.getBody();
SOAPElement requestElem = soapBody.addChildElement("getKnowledgeRequest",
"knowledge");
SOAPElement titleElem = requestElem.addChildElement("title",
"knowledge");
titleElem.addTextNode(title);

// SOAP Headers
MimeHeaders headers = soapMessage.getMimeHeaders();
headers.addHeader("SOAPAction", NAMESPACE  + "getKnowledgeRequest");

soapMessage.saveChanges();
```

Folgender Maßen sind wir mit dem Response des Servers umgegangen:

```
TransformerFactory transformerFactory = TransformerFactory.newInstance();
Transformer transformer = transformerFactory.newTransformer();
Source sourceContent = soapResponse.getSOAPPart().getContent();
System.out.print("\nResponse SOAP Message = ");
StreamResult result = new StreamResult(System.out);
transformer.transform(sourceContent, result);
```

3 Feedback

3.1 Probleme

Beim versuch die JavaScript teile der index.html mittels thymeleaf

auszuliefern, bin ich gescheitert, darum wurden dann die Javascript Teile in die main.js ausgelagert.

3.2 Zeitaufzeichnung

Hampl		**Wortha**	
Datum	**Zeit**	**Datum**	**Zeit**
11.12.2015	2 Stunden	11.12.2015	2 Stunden
13.12.2015	4 Stunden	18.12.2015	2 Stunden
17.12.2015	2 Stunden	03.01.2016	5 Stunden
18.12.2015	2 Stunden	06.01.2016	3 Stunden
03.01.2016	5 Stunden		
06.01.2016	3 Stunden		
Gesamt:	**18 Stunden**	**Gesamt:**	**12 Stunden**

4 Quellen

[1]: Producing a SOAP web service, Spring Dokumentation;
Online: https://spring.io/guides/gs/producing-web-service/
Zuletzt abgerufen: 18.11.2015

[2]: Building a RESTful Web Service, Spring Dokumentation;
Online: https://spring.io/guides/gs/rest-service/
Zuletzt abgerufen: 18.11.2015

[3]: Writing a Javascript REST client;

Online:http://blog.miguelgrinberg.com/post/writing-a-javascript-rest-client

Zuletzt abgerufen: 06.01.2016