UNIVERSITÄT SIEGEN

Fakultät IV
Betriebssysteme und verteilte Systeme
Prof. Dr. rer. nat. Roland Wismüller

Aufgabenblatt 5: Web Services

(Zu bearbeiten bis 31.01.2020)

Client/Server-Programmierung
Wintersemester 2019/20

Aufgabe 1: Hello World Web Service

Laden Sie sich zunächst den Code zum Aufgabenblatt von der CSP-Webseite herunter, entpacken Sie das Archiv und wechseln Sie ins Verzeichnis `HELLO`.

Das Beispiel verwendet die Axis2-Werkzeuge `java2wsdl.sh` und `wsdl2java.sh`, sowie das Hot Deployment, bei dem Sie das Java-Archiv des Services lediglich in das Axis2-Verzeichnis kopieren müssen.

Erstellen Sie den Web Service und führen Sie das Deployment durch, wie in der Vorlesung in Kapitel 7.4.2 beschrieben ist (verwenden Sie bitte **nicht** das beigefügte `makefile`). Erstellen Sie dann den Client. Überprüfen Sie nach jedem Schritt die neu generierten Dateien und finden Sie heraus, welche Rolle jede Datei spielt. Testen Sie den Web Service mit Ihrem Web-Browser und dem Java-Client.

Aufgabe 2: Börsenanwendung als Web Service

Implementieren Sie die Börsenanwendung aus dem Aufgabenblatt 1 in Form zweier Web Services unter Verwendung der Axis2 Werkzeuge. Da Web Services keine Erzeugung entfernter Objekte unterstützen, realisieren Sie das Depot wie bei der EJB-Realisierung als „Sitzungs-Objekt“, d.h., verwenden Sie die von Axis2 angebotene Sitzungsverwaltung mit Hilfe von HTTP Cookies bzw. SOAP, wie in Kapitel 7.4.4 der Vorlesung beschrieben.

Gehen Sie wie im Tutorial beschrieben vor, d.h., erstellen Sie zunächst Java-Schnittstellen, aus denen Sie die WSDL-Dokumente erzeugen. Erstellen Sie dann zu den WSDL-Beschreibungen die Implementierungen der Web Services und anschließend den Client. Testen Sie anschließend Ihre Implementierung mit mindestens zwei Clients.

Hinweise:

- Für das *Data Binding* sollten Sie bei `wsdl2java.sh` am besten `adb` verwenden.
- Sehen Sie sich vor der Implementierung die erzeugte WSDL-Schnittstelle an, bzw. die von `wsdl2java.sh` daraus generierten Methodensignaturen. Ggf. müssen Sie Parameter- oder Rückgabetypen anpassen.
- Da es in Axis2 nicht (oder zumindest nicht ohne weiteres) möglich ist, eine lokale Referenz auf einen Service zu erhalten, sollten Sie in der `Depot`-Implementierung eine Neue Instanz der `StockExchange`-Implementierung erzeugen, um deren Dienste nutzen zu können.
- Sie können beide Dienste in ein gemeinsames Archiv packen und zusammen deployen, wenn Sie sich die jeweils von `wsdl2java.sh` erzeugten Service-Deskriptoren (`services.xml`) kopieren und anschließend zu einem einzigen Deskriptor zusammenfassen.
- Zumindest wenn Sie `adb` als *Data Binding* verwenden, werden Klassen für komplexe Parameter- und Ergebnistypen (konkret: `WinnerInfo` und `DepotEntry`) als innere Klassen der jeweiligen Stubklassen erzeugt (also z.B. `StockExchangeStub.WinnerInfo`)
- Wenn `wsdl2java.sh` mit `adb` eine Warnung „Type http://ws.apache.org/axis2StockException missing!“ ausgibt, dürfen Sie diese ignorieren.

Zusatz (für Motivierte):
Untersuchen Sie die Kommunikation zwischen Client und Web Service mit Hilfe des TCP Monitors genauer, den Sie unter `/opt/dist/tcpmon-1.0/bin/tcpmon.sh` finden. Der TCP Monitor bietet die Möglichkeit, die an einen Web Service gesendeten und vom Web Service empfangenen SOAP Nachrichten im Klartext einzusehen (siehe Web Service Tutorium). Untersuchen Sie dabei insbesondere, wie die Sitzungsverfolgung funktioniert.

Aufgabe 3: Börsenanwendung als EJB Web Service (für Motivierte)

Machen Sie aus Ihrer EJB für `StockExchange` von Aufgabenblatt 4 mit Hilfe der Annotation `@WebService` einen Web Service. Da EJB keine Sitzungsverwaltung für Web Services vorsieht, kann ein Web Service nur aus einer *stateless* Session Bean erzeugt werden. Daher müssen Sie das Depot wie in Aufgabe 3 von Blatt 4 mit Hilfe von Entities realisieren.

Erstellen Sie für diesen Web Service dann einen Client mit Hilfe der Axis2 Werkzeuge.