

Praktikum "Daten- und Systemintegration" – WS 2009/2010

## Praktikumsaufgabe 2

**Labor : WI-Labor D14/1.11**

**Praktikumstermine** (x-Raster, Fr 3. Block)

**Gruppen:** 1 Praktikumsgruppe mit max. 16 Teilnehmern  
in 2er-Teams

Prof. Dr. F. Bühler, Prof. Dr. G. Turetschek  
Praktikumsunterstützung: Jan Horneck (B.Sc.)

## Inhaltsverzeichnis

1	Praktikumsaufgabe.....	3
1.1	Testen der bestehenden Web-Services.....	4
1.2	Einloggen auf WSO2 ESB.....	8
1.3	Einführung in die Management-Console.....	10
1.4	Erstellung des Service Proxies.....	15
1.4.1	Erstellung des Endpoints.....	15
1.4.2	Erstellung des Proxies.....	17
1.4.3	Testen des Proxies aus SoapUI .....	23
1.5	Hinzufügen der Authentifizierung.....	25
1.5.1	Der Ablauf der Mediation.....	25
1.5.2	Bearbeiten des Proxy Services.....	26
1.6	Editieren der Datenbank-Einträge.....	30
1.7	Testen der Authentifizierung in SoapUI.....	34
1.8	Testen der Protokollierung in SoapUI.....	35
1.9	Modifikation des WS-Client Programms.....	37
1.9.1	Vorbereitungen.....	37
1.9.2	Löschen des BestellBean Web-Service-Clients.....	37
1.9.3	Erstellen des BestellBeanProxy-Web-Service-Clients.....	38
1.9.4	Anpassung der Methoden in Bestellung.java.....	39
1.9.5	Setzen der Authentifizierungs-Daten.....	40
1.9.6	Ausführung von WS-Client.....	41
2	Abgabe.....	42

# 1 Praktikumsaufgabe

Im zweiten Praktikum werden fortgeschrittene Techniken, die im Rahmen eine SOA Verwendung finden, eingesetzt. So sollen Sie den Umgang mit dem Produkt *WSO<sub>2</sub>ESB* erlernen.

Mithilfe von *WSO<sub>2</sub>ESB* – welches auf dem WMS *Synapse* basiert - soll ein Web-Service mit einer Authentifizierung versehen werden, so dass nur Clients, welche den Namen und das Passwort kennen, den Web-Service nutzen können. Außerdem werden die Zugriffe auf den Web-Service in einer Datenbank protokolliert.

Um die Authentifizierung zu erreichen, muss das WMS entsprechend konfiguriert werden und ein *Web-Service-Proxy* eingerichtet werden. Des weiteren muss die im Praktikum 1 entwickelte Client-Anwendung angepasst werden, damit diese die Authentifizierungs-Daten (HTTP-Authentication) an das WMS sendet.

Um die Funktionsfähigkeit der einzelnen Web-Services zu überprüfen, wird das Werkzeug *SoapUI* verwendet, welches viele Funktionen zum Testen von Web-Services besitzt.

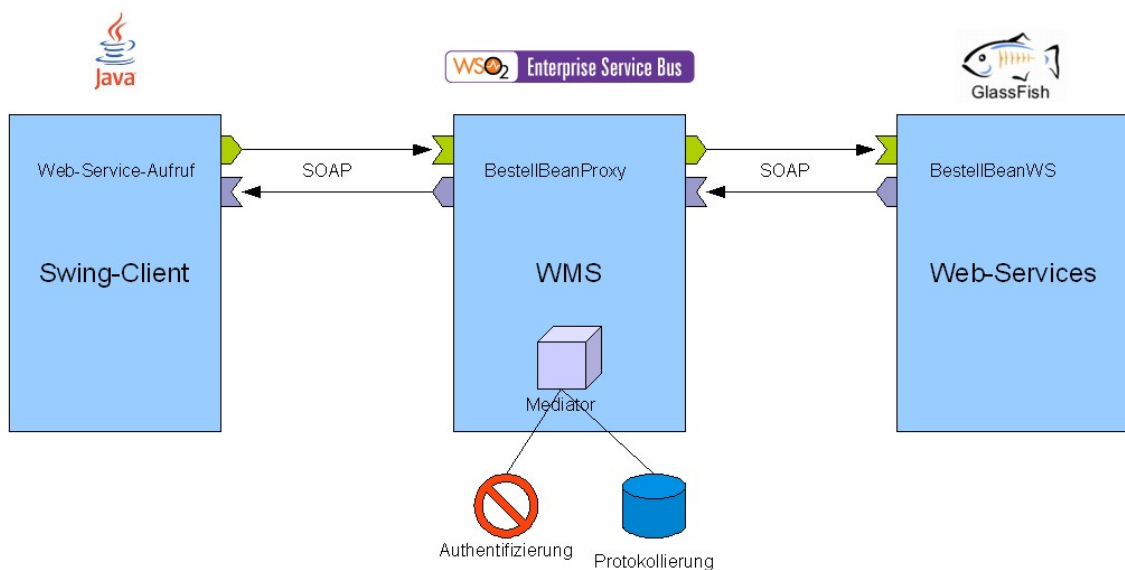
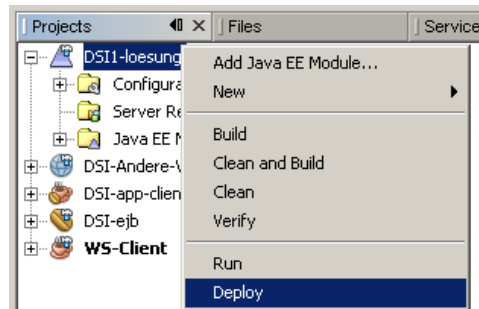


Abbildung 1: Aufbau der Praktikumsaufgabe

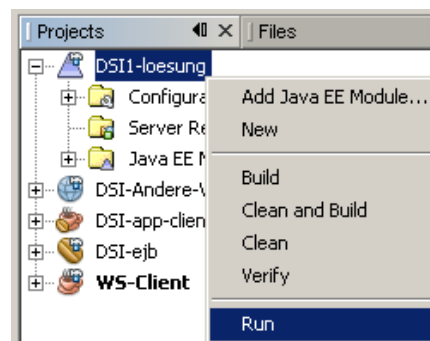
## 1.1 Testen der bestehenden Web-Services

Als erstes sollen die im letzten Praktikum erstellten Web-Services mit dem Werkzeug *SoapUI*<sup>1</sup> getestet werden. Starten sie dazu *Netbeans* und öffnen Sie die bereitgestellte Musterlösung (Projekt **DSI1-loesung**, die Checkbox **Open Required Projects** muss aktiviert sein).

Zunächst starten Sie den Applikationsserver *Glassfish* (Fenster: **Services** → **GlassFish v2** → **Start**). Anschließend kompilieren (Kontextmenü **Clean and Build**) und *deployen* (Kontextmenü **Deploy**) Sie die Enterprise Application **DSI1-loesung** auf den Applikationsserver.



Starten sie den EJB-Client mittels einem Rechtsklick auf **DSI1-loesung** und der Auswahl von **Run** aus dem Kontext-Menu.



Erstellen Sie – wie bereits im ersten Praktikum - die Demo-Daten. Nach dem Deployment minimieren Sie nun das *Netbeans*-Fenster (N.B. Sie dürfen das Fenster nicht schließen, da sonst auch *Glassfish* beendet wird).

Starten Sie als nächstes *SoapUI*. Ein Icon hierfür sollte sich auf dem Desktop befinden. Falls nicht ist es im Explorer über die Batch-Datei **C:\Programme\evaware\soapUI-2.0.2\bin\soapui.bat** zu starten.

Nach dem Starten von *SoapUI* öffnet sich folgendes Fenster (siehe Abbildung 2: Standard-Ansicht von SoapUI).

---

<sup>1</sup> SoapUI gibt es in einer Open-Source und einer Kommerziellen Version (mit zusätzlichen Funktionen). Es kann auch mittels Java-Webstart direkt aus dem Browser heraus gestartet werden.  
Homepage: <http://www.soapui.org/> Stand: 2009-01-15

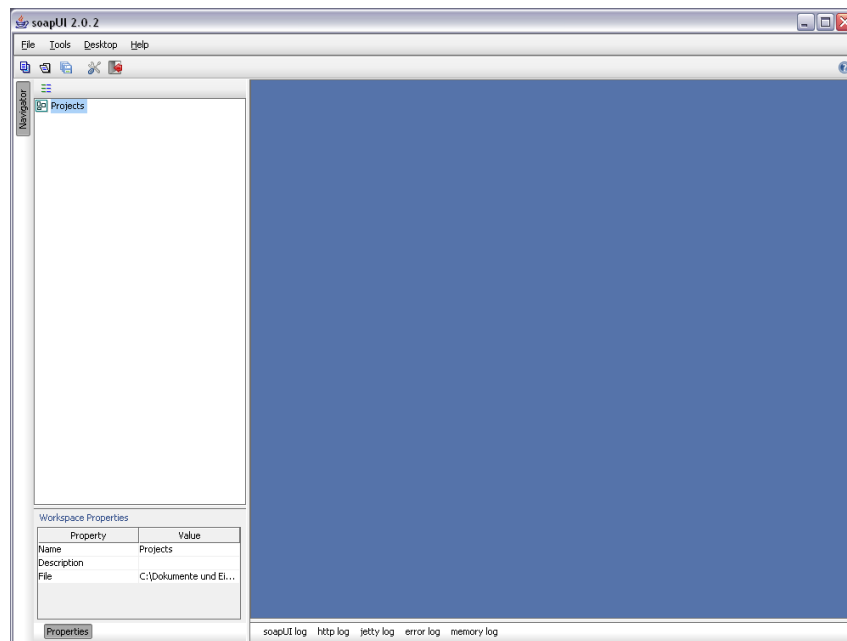


Abbildung 2: Standard-Ansicht von SoapUI

Als nächstes soll ein Projekt zum Testen des **BestellBean-Web-Services** angelegt werden. Um das Projekt anzulegen, wird die URL der WSDL-Datei des Web-Services benötigt. Diese kann ermittelt werden indem in *Netbeans* im Projekt *DSI-ejb* ( DSI-ejb ) **Test Web Service** aus dem Kontext-Menue von **BestellBeanService** aufgerufen wird (siehe Abbildung 3).

Auf der sich öffnenden Webseite befindet sich neben den Eingabe-Formularen zum Testen des Web-Services auch ein Link, zu dessen WSDL-Datei (siehe Abbildung 4). Kopieren Sie die URL (= Link-Adresse) in die Zwischenablage<sup>2</sup>.

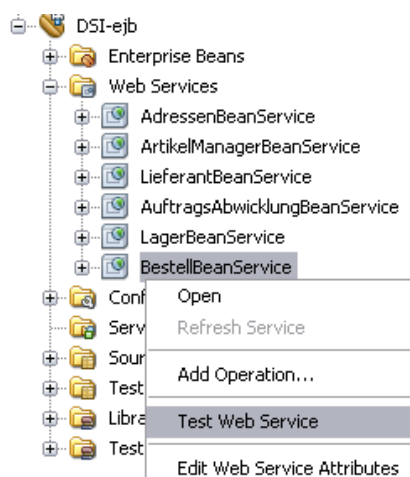


Abbildung 3: Testen eines Web-Services

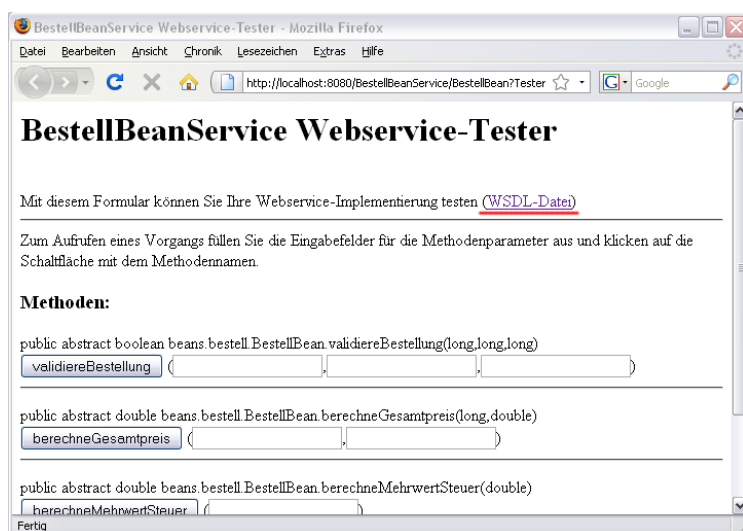


Abbildung 4: Link zur WSDL-Datei

Wählen Sie nun in der Menue-Leiste von *SoapUI* **File** → **New WSDL Project** aus, und fügen Sie in dem sich öffnenden Dialog unter **Initial WSDL** die URL der WSDL-Datei ein. Als **Project**

<sup>2</sup> Es sollte sich um Folgende URL handeln: <http://localhost:8080/BestellBeanService/BestellBean?WSDL>

**Name** bietet sich *BestellBeanDirect* an. Aktivieren Sie die Checkbox **Create Project File** (siehe Abbildung 5)

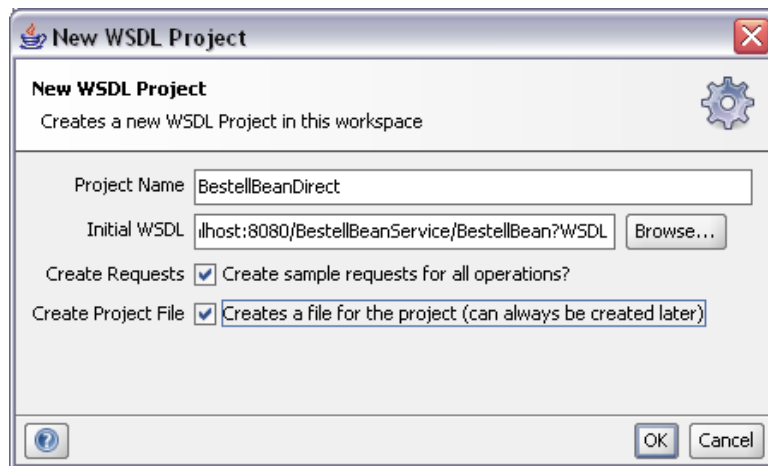



Abbildung 5: Dialog zur Erstellung eines neuen WSDL-Projekts

Speichern Sie die Projektdaten-Datei (**BestellBeanDirect-soapui-project.xml**) auf der Festplatte ab (Laufwerk **D:** oder Laufwerk **Z:**). *SoapUI* generiert ein Projekt, welches für jede Operation des Web-Services eine SOAP-Request-Nachricht bereitstellt.

Durch Auswahl einer Nachricht in der Projekt-Ansicht wird im rechten Bereich von *SoapUI* ein Fenster geöffnet. Dieses ist in der Mitte getrennt, und zeigt im linken Eingabefeld die Request-Nachricht an. Die *Request-Nachricht* lässt sich bearbeiten und mittels dem Knopf mit dem grünen Pfeilsymbol (  ) an den Web-Service senden. Die *Response-Nachricht* des Web-Services wird schließlich im rechten Textfeld angezeigt.

Im Bereich unter der Projekt-Ansicht lassen sich Einstellungen zur *Request-Nachricht* vornehmen. Hier wird später der Benutzername und das Passwort für die Authentifizierung gesetzt.

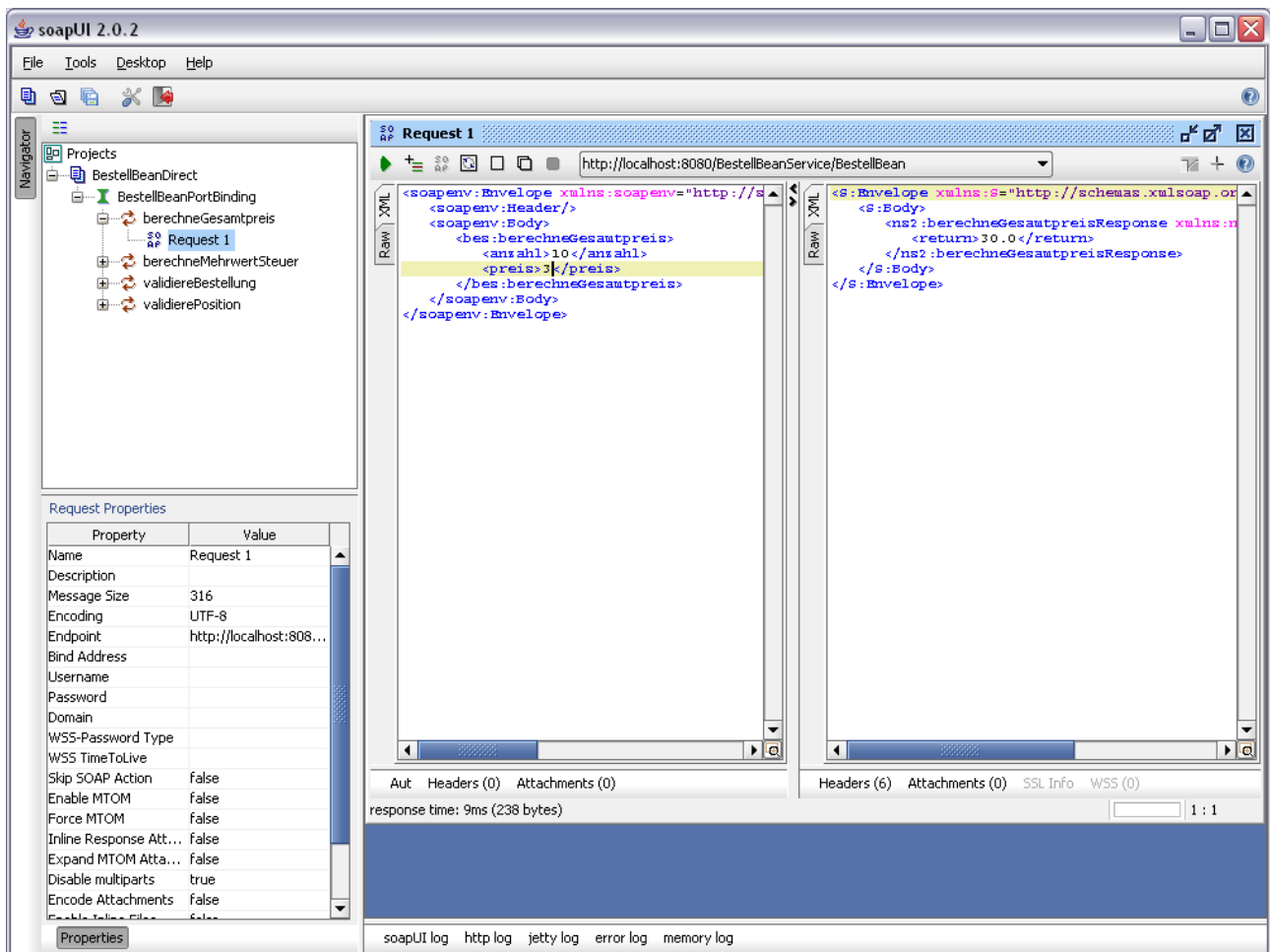


Abbildung 6: Typische SoapUI Ansicht. Links-Oben befindet sich die Projekt-Ansicht. Rechts befindet sich ein Editor für die ausgewählte Request-Nachricht, sowie wird deren zugehörige Response-Nachricht angezeigt. Links-Unten lassen sich Einstellungen zur Request-Nachricht vornehmen. Rechts-Unten lassen sich verschiedene Logs anzeigen.

Geben Sie bei der Request-Nachricht im linken Bereich des Request-Editors anstelle der Fragezeichen entsprechende Test-Werte ein. Prüfen Sie, dass der Web-Service wie erwartet funktioniert.

## 1.2 Einloggen auf WSO2 ESB

Als nächstes soll mit Hilfe von *WSO<sub>2</sub> ESB* ein Proxy-WebService für den im 1. Praktikum entwickelten BestellBean-Web-Service zu erzeugt werden. Mit Hilfe eines bereits entwickelten Mediators soll die Authentifizierungsfunktion realisiert werden.

*WSO<sub>2</sub> ESB* wird über einen Server bereitgestellt welcher die IP-Adresse 141.100.41.222 hat. Für jedes Praktikumsteam existiert auf diesem Server eine eigene Installation. Dementsprechend ist es wichtig, dass Sie im folgenden Teil des Praktikums die richtigen Ports verwenden.

Für jeden Port benötigen Sie die Nummer Ihres Praktikumsteams welche Sie beim ersten Praktikums-Termin erhalten haben.

Der Basis-Port berechnet sich folgendermaßen:

$$\text{Basis-Port} = 60000 + \text{Nummer} * 100$$

Daraus berechnen sich die benötigten Ports wie folgt:

Name	Port	Beschreibung
Management-Konsolen-Port	Basis-Port + 14	Über diesen Port kann man sich in die Management-Konsole einloggen. Die URL hierfür ist <a href="https://141.100.41.222:&lt;Port&gt;/esb">https://141.100.41.222:&lt;Port&gt;/esb</a>
Java-DB-Port	Basis-Port + 15	Über diesen Port kann man sich an die JavaDB-Datenbank anmelden, welche für die Authentifizierung benötigt wird. Die Database-URL lautet: <a href="jdbc:derby://141.100.41.222:&lt;Port&gt;/ewms">jdbc:derby://141.100.41.222:&lt;Port&gt;/ewms</a>
SOAP-Port	Basis-Port + 12	Über diesen Port kann auf den zu erstellenden Web-Service-Proxy zugegriffen werden. Die erstellten Proxies lassen sich über folgende URL anzeigen: <a href="http://141.100.41.222:&lt;Port&gt;/soap/">http://141.100.41.222:&lt;Port&gt;/soap/</a>



Bitte bedenken Sie dass die Screenshots der Management-Konsole und Soap-UI auf eine Installation von *WSO<sub>2</sub>Esb* auf den lokalen Rechner beziehen. Die URLs müssen in diesem Fall angepasst werden. (localhost -> 141.100.41.222, Ports beachten)

Loggen Sie sich nun in die Management-Konsole von *WSO<sub>2</sub> ESB* ein.

URL	<a href="https://141.100.41.222:&lt;Port&gt;/esb/">https://141.100.41.222:&lt;Port&gt;/esb/</a>
Username	admin
Password	admin



Übersehen Sie nicht, dass die URL mit **https** beginnt.





Einige Funktionen der Management-Konsole sind nicht mit *Firefox 3.5* kompatibel.

Verwenden Sie den bereitgestellten *Internet-Explorer 8* oder installieren Sie bitte *Firefox 3.0* als Portable-App auf Laufwerk **D:\**.  
Er kann von der Website

[http://portableapps.com/apps/internet/firefox\\_portable/localization#legacy](http://portableapps.com/apps/internet/firefox_portable/localization#legacy) heruntergeladen werden.

Da das Zertifikat des Servers selbst-signiert ist, müssen sie eine Sicherheits-Ausnahme hinzufügen.



Abbildung 7: Hinzufügen einer Sicherheits-Ausnahme.

Nach der Bestätigung der Ausnahme erreichen sie schließlich die Login-Seite.

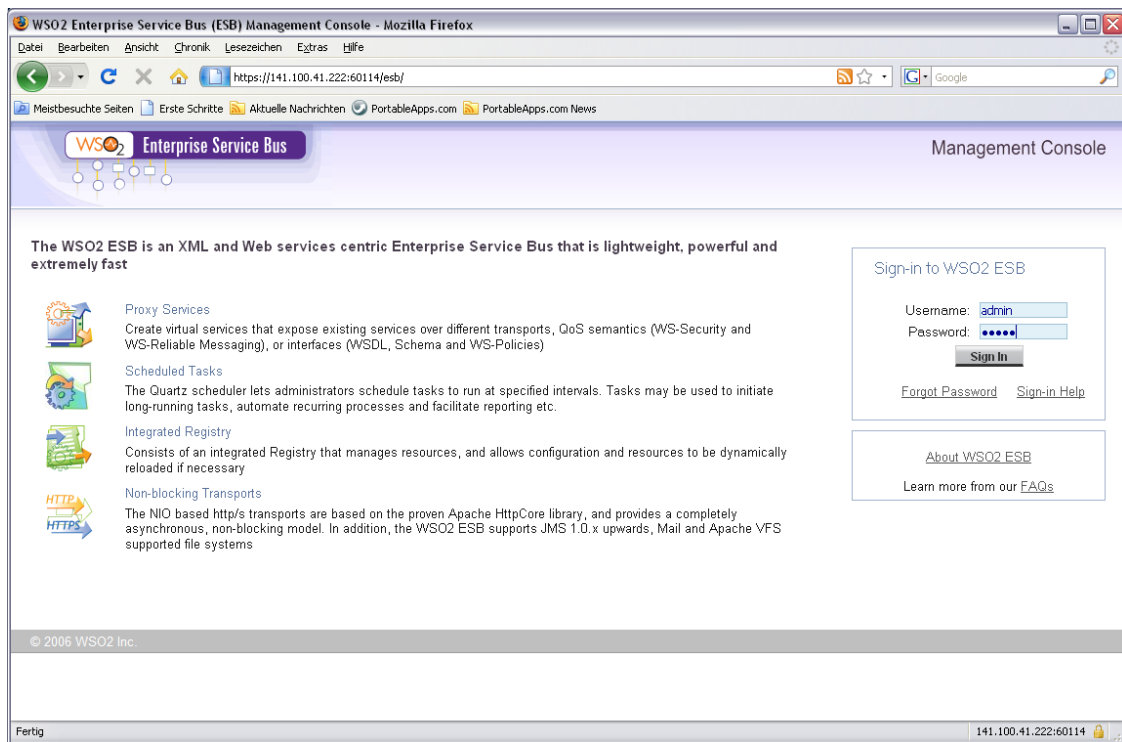


Abbildung 8: Login-Seite von WSO<sub>2</sub> ESB (name: admin, pw: admin)

## 1.3 Einführung in die Management-Console

Nach dem Login befindet man sich in der *Management-Console* welche folgenden Aufbau hat (siehe Abbildung 9).

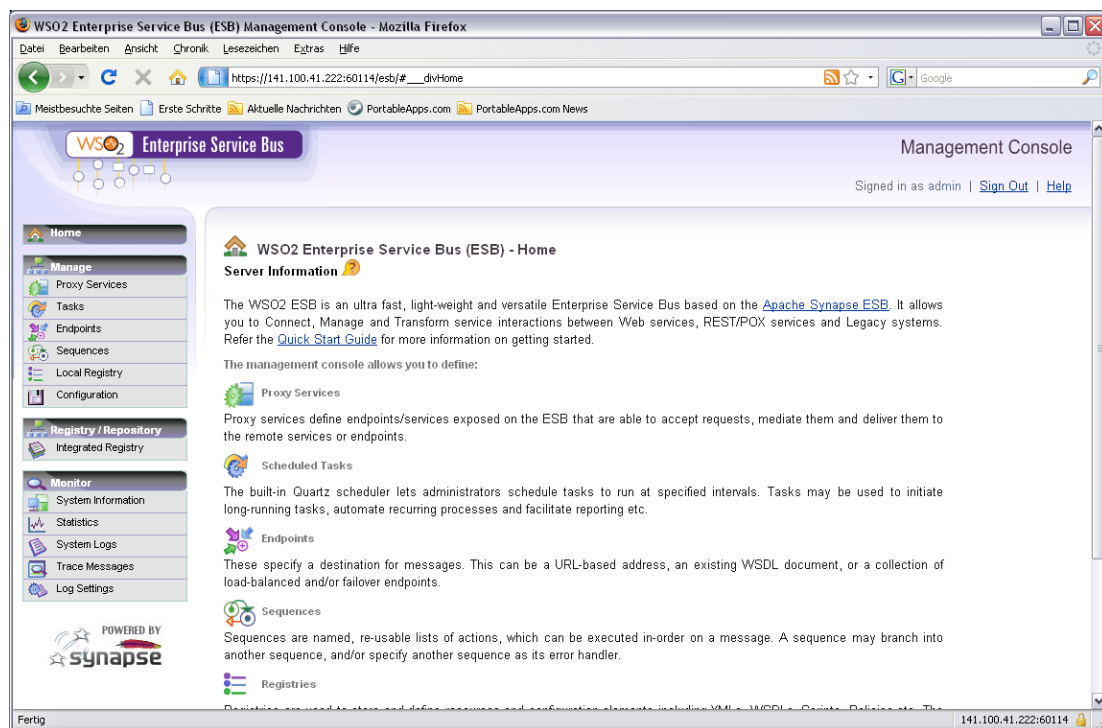


Abbildung 9: Start-Bildschirm der Management-Console.

In der linken Navigationsleiste sind eine Reihe von Menue-Punkten vorhanden. Unter dem Punkt **Manage** → **Configuration** bekommt man eine Ansicht der momentan generierten **synapse.xml**-Datei (siehe Abbildung 10). Diese Datei wird letztendlich von *Apache Synapse* als Konfigurationsdatei verwendet. *WSO<sub>2</sub> ESB* ist sozusagen ein grafischer Editor für diese XML-Datei. Diese Seite bietet einem die Möglichkeit den XML-Quelltext direkt zu bearbeiten, muss aber auch besucht werden, nachdem Änderungen mittels der Management-Console gemacht wurden. Nach dem durchführen von Änderungen ist hier der **Save**-Button zu drücken.

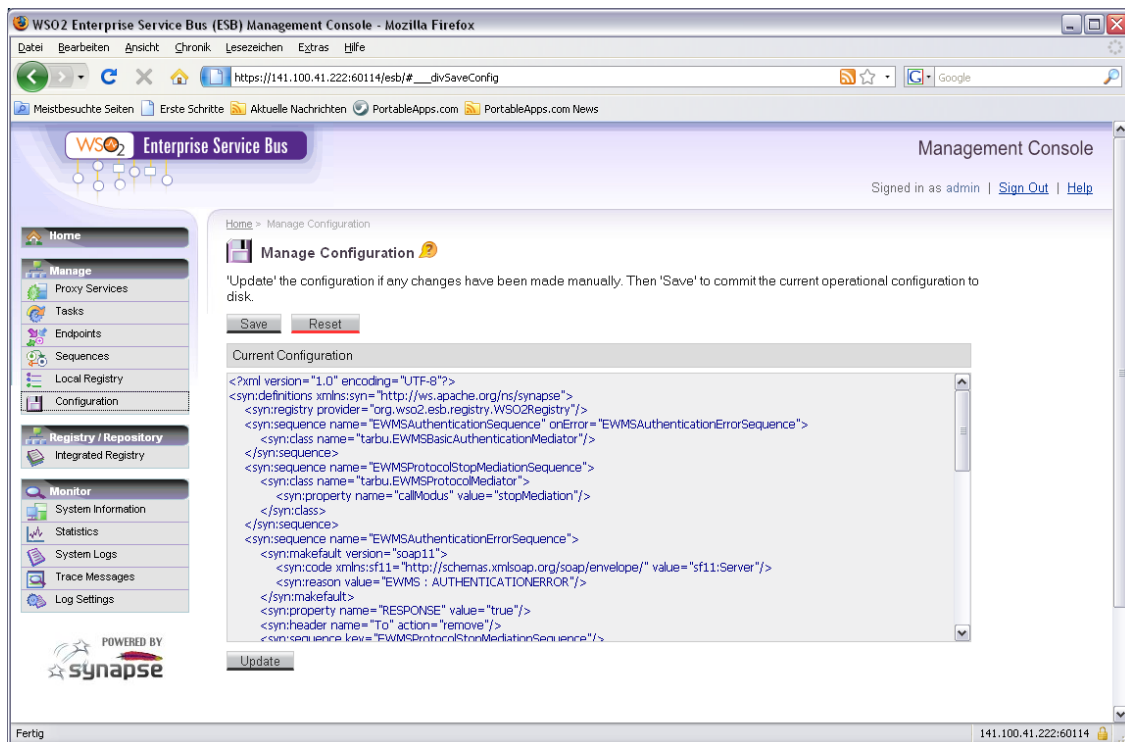


Abbildung 10: WSO<sub>2</sub> ESB: Manage Configuration

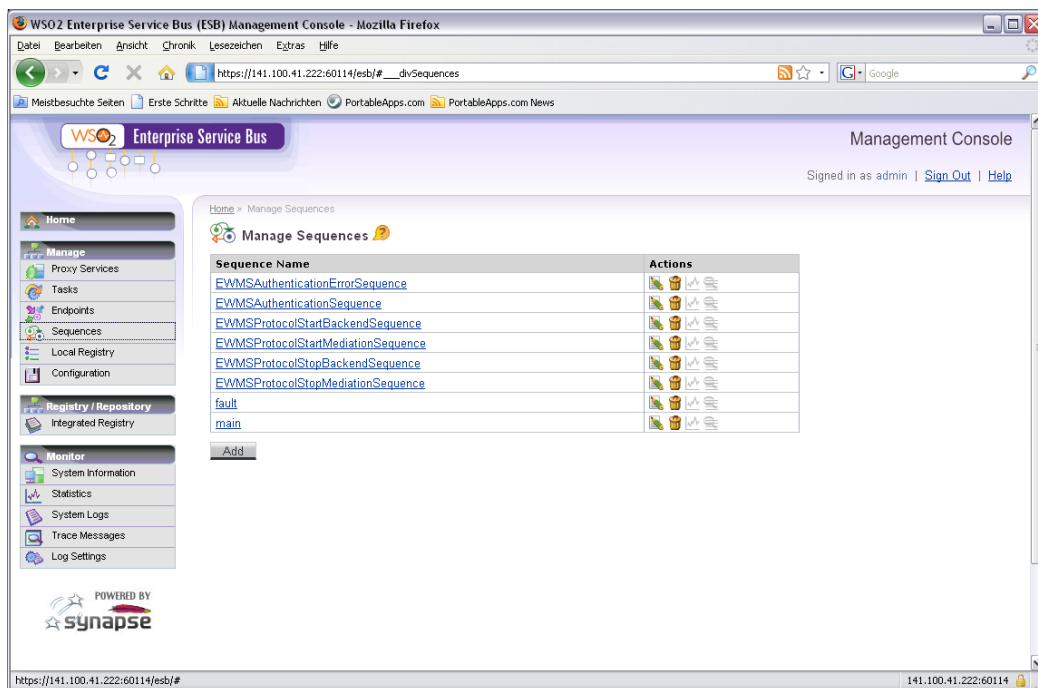


Abbildung 11: WSO<sub>2</sub> ESB: Manage Sequences

Unter dem Punkt **Manage** → **Sequences** werden Sequenzen – d.h. Folgen von hintereinander auszuführenden Mediatoren – verwaltet (siehe Abbildung 11). Die Sequenzen **fault** und **main** sind Standard-mäßig vorhanden, die anderen Sequenzen wurden bereits vorbereitet, und werden später bei der Authentifizierung verwendet. Sequenzen sind Blöcke welche mit einem Namen versehen sind, unter dem sie an anderen Stellen wiederverwendet werden können.

Unter dem Punkt **Manage** → **Endpoints** befindet sich die Verwaltung von Web-Service-Endpoints. Ein Web-Service-Endpoint stellt die Adresse eines Web-Services dar. Über die definierten Endpoints kann aus *WSO<sub>2</sub> ESB* auf Web-Services zugegriffen werden. Zur Zeit sind noch keine Endpoints vorhanden (siehe Abbildung 12).

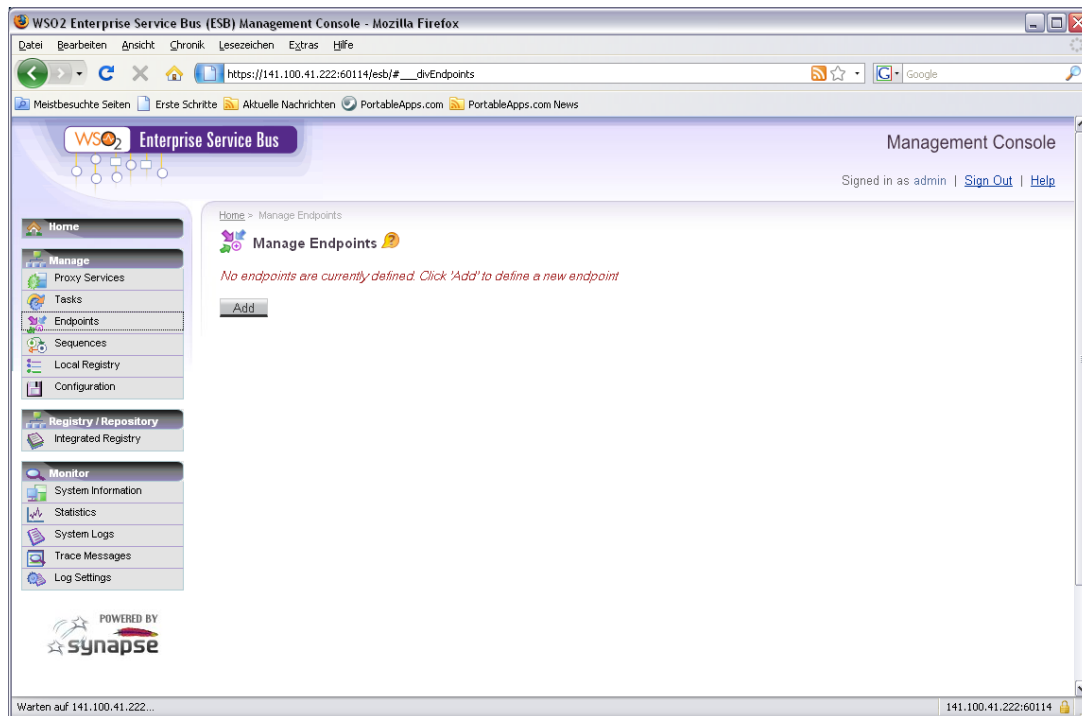


Abbildung 12: WSO<sub>2</sub> ESB: Web-Service Endpoints

Unter dem Punkt **Manage** → **Proxy Services** sind schließlich die erstellten Web-Service-Proxies zu finden. Ein Proxy-Service ist ein Web-Service den *WSO<sub>2</sub> ESB* bereitstellt, und bei dessen Aufruf einige Mediatoren durchlaufen werden, und schließlich ein anderer Web-Service aufgerufen wird. Auch hier ist zur Zeit noch kein Proxy vorhanden (siehe Abbildung 13).

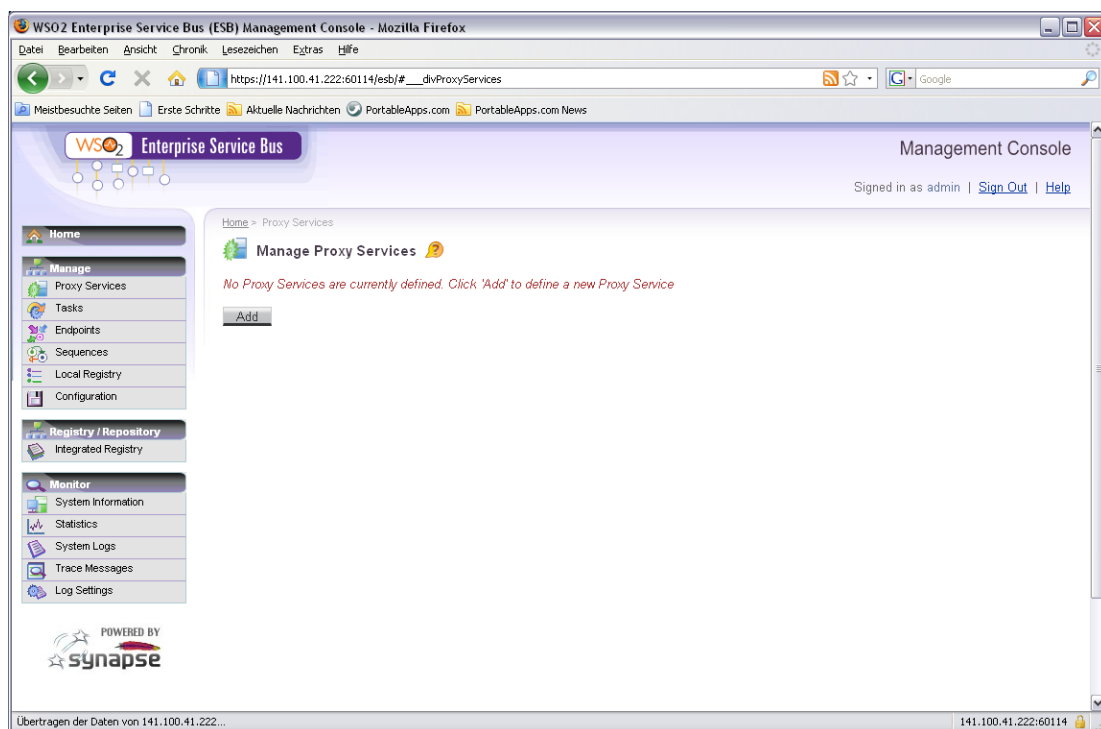


Abbildung 13: WSO<sub>2</sub> ESB: Manage Proxy Services

Unter **Monitor** → **System Logs** kann schließlich die Log-Datei von WSO<sub>2</sub> ESB betrachtet werden (siehe Abbildung 28).

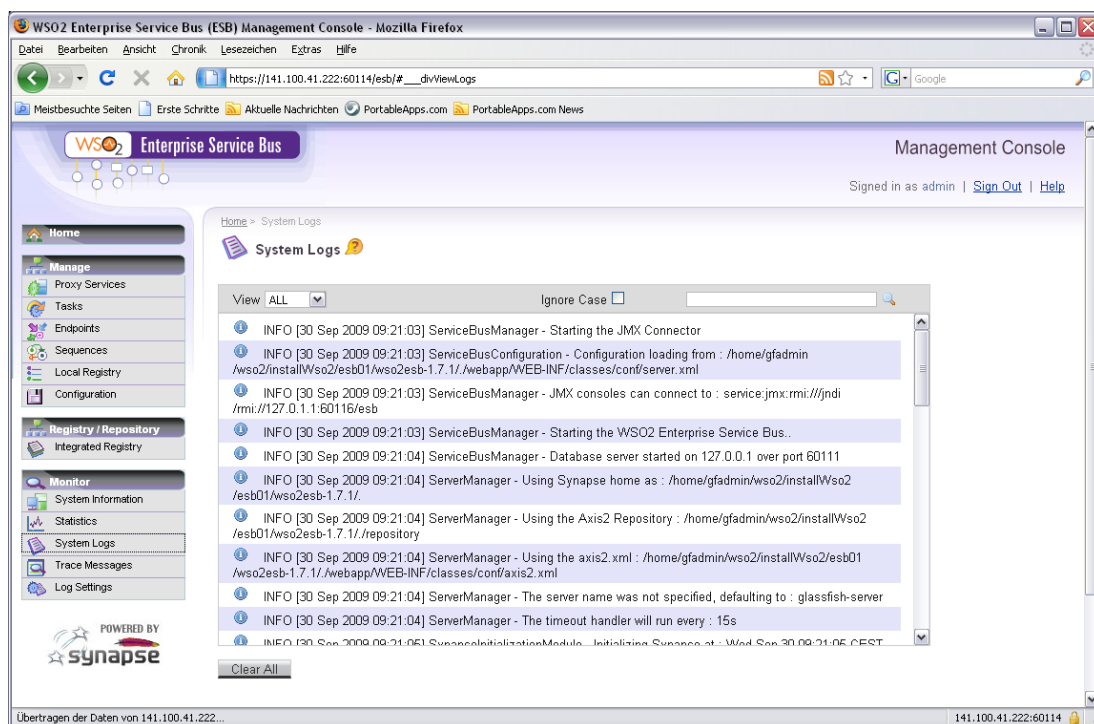


Abbildung 14: WSO<sub>2</sub> ESB: System Logs

## 1.4 Erstellung des Service Proxies

Im Rahmen dieser Praktikumsaufgabe soll ein Service Proxy für den *BestellBean-Web-Service* erstellt werden. Bei Aufruf des Service-Proxies, welchen das WMS bereitstellt, wird von diesem der ursprüngliche Web-Service aufgerufen. Der Proxy soll später so erweitert werden, dass er nur noch die Anfragen authentifizierter Clients verarbeitet, und außerdem sämtliche Anfragen in einer Datenbank loggt.

### 1.4.1 Erstellung des Endpoints

Zunächst muss ein *Endpoint* für den durch den Proxy zu kapselnden Web-Service erstellt werden. Hierzu ist in der Management-Console der Punkt **Manage** → **Endpoints** zu wählen. Auf diesem Formular ist der **Add**-Button zu drücken, und im Popup-Menue der Punkt **Address Endpoint** zu wählen (siehe Abbildung 15).

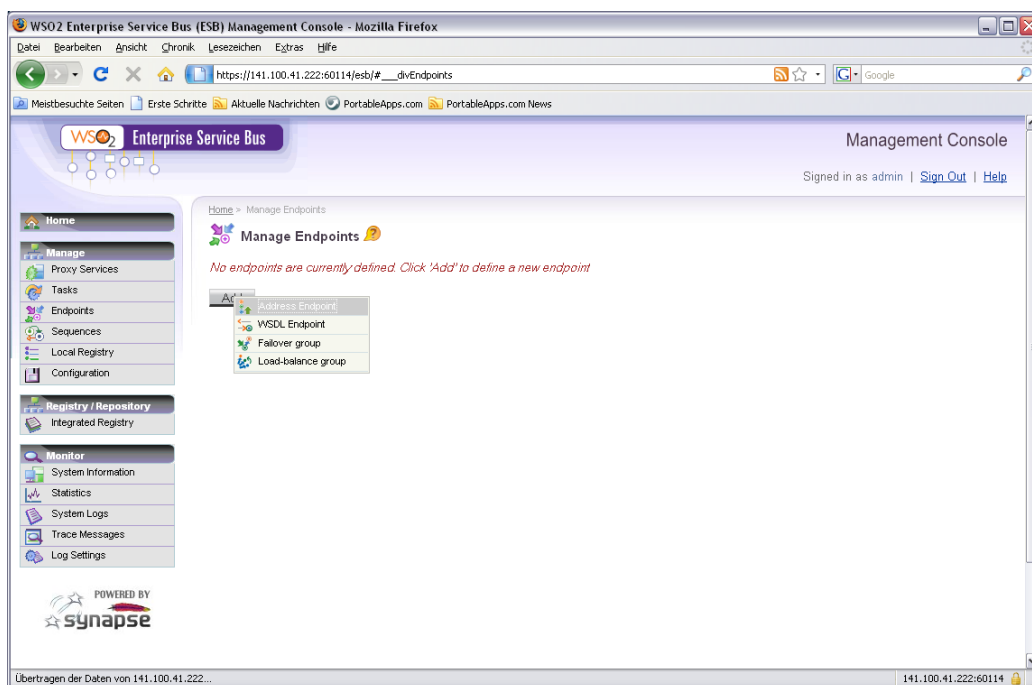


Abbildung 15: Erstellen eines Address Endpoints

In dem Formular ist als Name **BestellBeanWS** und die Adresse des Web-Services (nicht die dessen WSDL-Datei) anzugeben. Die Adresse kann **beinahe** wie üblich über Netbeans (Rechtsklick auf Web-Service → Test Web Service) festgestellt werden.

Doch hier gibt es einen wichtigen Punkt zu beachten! Standardmäßig wird in Netbeans der Web-Service mittels **localhost** als Server aufgerufen. Da sich WSO2ESB allerdings auf einem anderen Server befindet, kann von dort aus der Web-Service nicht über localhost aufgerufen werden.

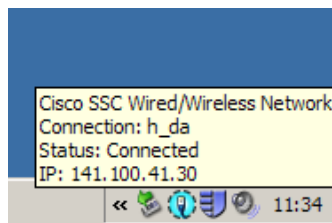


Abbildung 16: IP-Adresse des Rechners.

Hierfür ist im Browser-Fenster im folgenden bei der Angabe der URLs localhost durch die IP-Adresse des Praktikums-Rechners zu ersetzen. Am einfachsten kann diese ermittelt werden, indem man den Mauszeiger über das Netzwerk-Kabel-Icon bewegt, und die Adresse aus dem erscheinenden Tooltip abliest (siehe Abbildung 16). Alternative Möglichkeiten sind <sup>3</sup> und <sup>4</sup>.

Sie sollte <http://<ihre-ip-adresse>:8080/BestellBeanService/BestellBean> lauten. Danach ist der **Save**-Button zu drücken (siehe Abbildung 17). Jetzt wird in der **Manage** → **Endpoints** Ansicht der erstellte Endpoint angezeigt (siehe Abbildung 18).



Diese Änderung ist noch nicht wirklich festgeschrieben. Dies geschieht erst durch den Aufruf von **Manage** → **Configuration** und das Drücken des dortigen **Save**-Buttons. Ansonsten sind die Änderungen nach einem Neustart von *WSO<sub>2</sub> ESB* verloren.

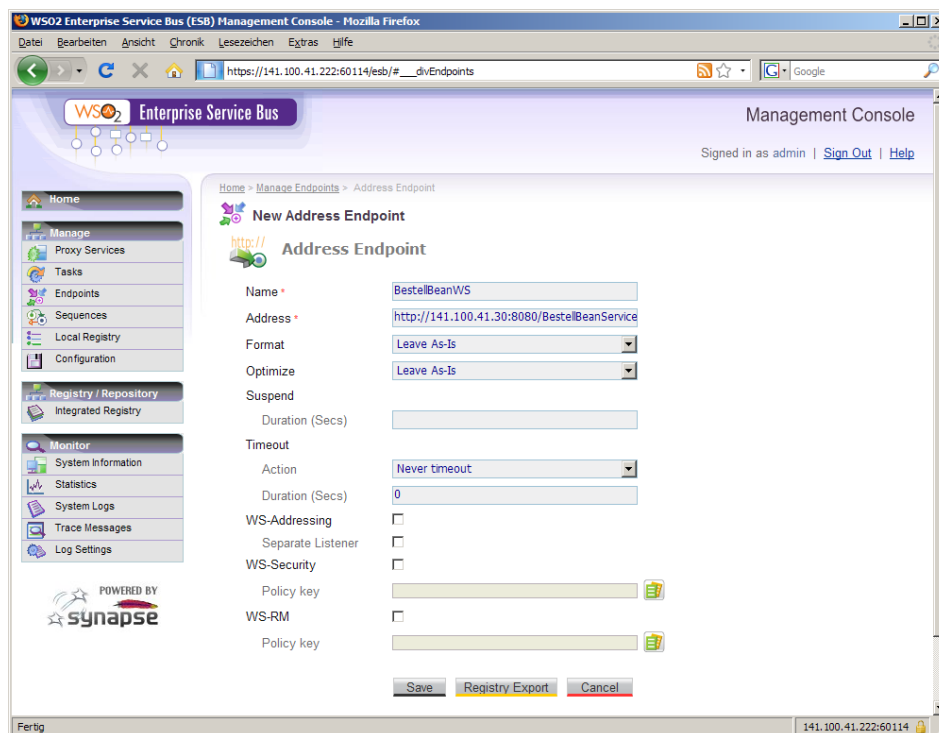


Abbildung 17: Daten für den Address Endpoint

<sup>3</sup> Diese kann beispielsweise über folgende URL bestimmt werden: <http://www.heise.de/netze/tools/ip/>.

<sup>4</sup> Alternativ dazu kann das Programm *ipconfig* aus der Eingabeaufforderung heraus verwendet werden.



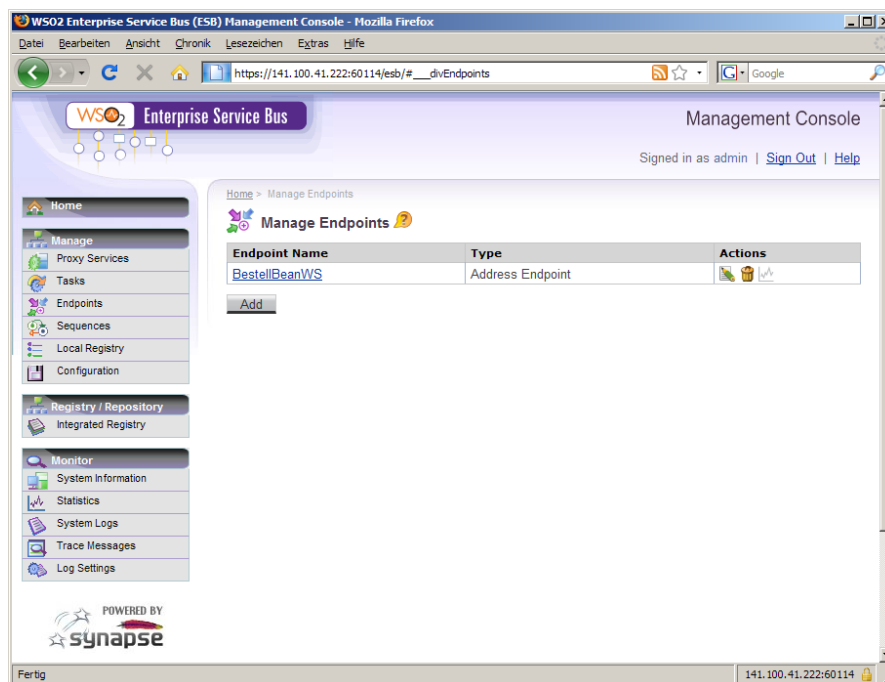


Abbildung 18: Der soeben erstellte Endpoint

### 1.4.2 Erstellung des Proxies

Der nächste Schritt ist das Erstellen des Proxies. Zur Eingabe der hierfür nötigen Daten, gelangt man über den Punkt **Manage** → **Proxy Services** und das Drücken des **Add**-Buttons.

Als Proxy Service Name ist **BestellBeanProxy** einzugeben. Bei *Target Endpoint* ist über das Pop-up-Menue - welches sich durch Drücken auf den Pfeil-Button öffnet – der erstellte Endpoint **Declared Endpoint** → **BestellBeanWS** zu wählen (siehe Abbildung 19).

Die 3 *Target-Sequences* sind die Sequenzen welche die ein- bzw. ausgehenden Nachrichten durchlaufen, bzw. die Sequenz welche im Fehlerfall durchlaufen wird. Hier können entweder vorher definierte Sequenzen gewählt werden, oder die Sequenzen als anonym spezifiziert werden, was bedeutet, dass diese keine Namen besitzen.

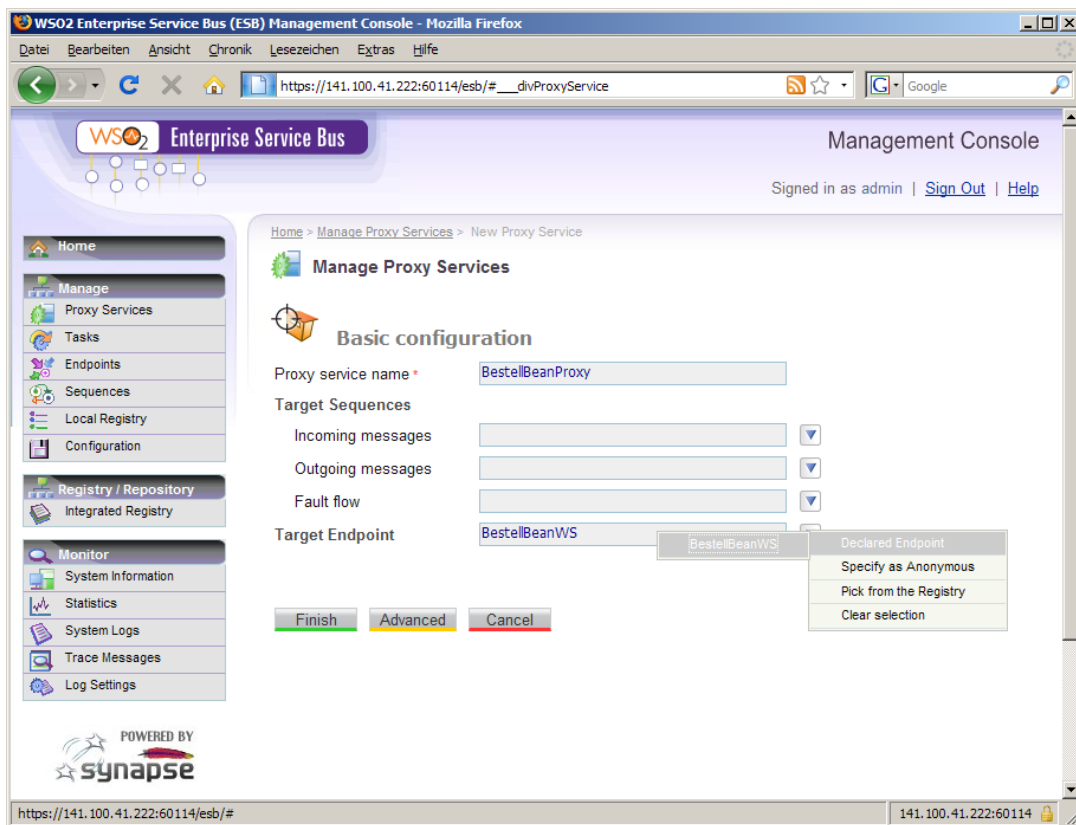


Abbildung 19: Erstellung des Proxy Services

Für den vorläufigen Proxy wird nur eine Sequenz für die **Outgoing messages** benötigt. Der Einfachheit halber sollte diese mittels **Specify as Anonymous** erstellt werden. Alternativ wäre es auch möglich diese *Sequence* unter **Manage** → **Sequences** zu erstellen und mit einem Namen zu versehen. Da diese Sequenz jedoch nur an einer Stelle verwendet wird, und nicht allzu komplex ist, sollte dieser zusätzliche Schritt lieber gespart werden.

Zu dieser *Sequence* ist lediglich ein **Send-Mediator** hinzuzufügen. Dies geschieht über das Drücken des **Add Mediator**-Buttons und die Auswahl von **Core** → **Send** im Popup-Menu (siehe Abbildung 20). Danach muss der **Save**-Button gedrückt werden.

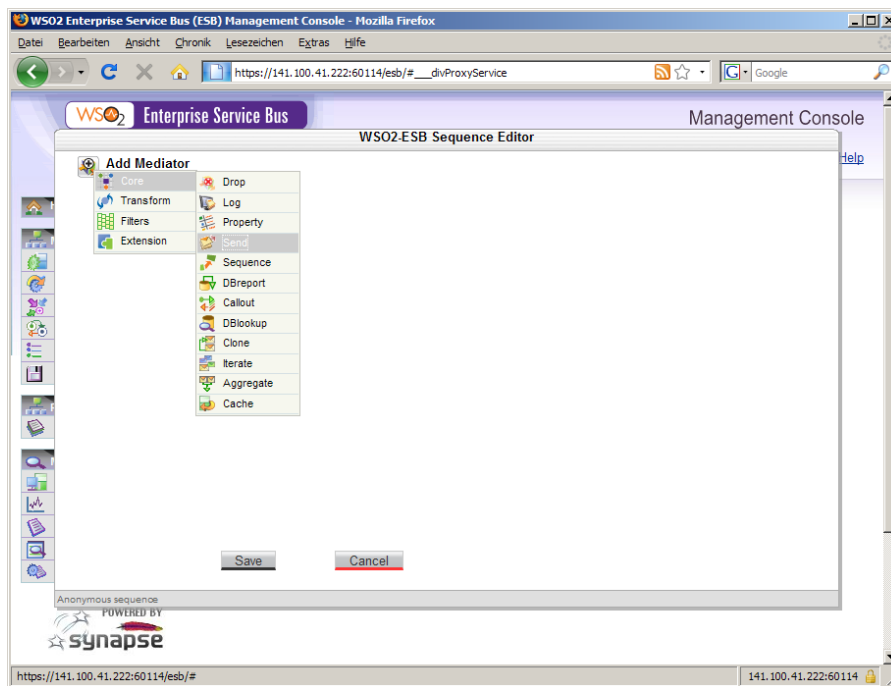


Abbildung 20: Hinzufügen des Send-Mediators zur Sequence.

Nun wird die **Anonymous Sequence** bei **Outgoing message** aufgelistet (siehe Abbildung 21). Bevor der Proxy fertiggestellt ist, müssen noch einige Einstellungen vorgenommen werden. Hierzu muss der **Advanced**-Button betätigt werden.

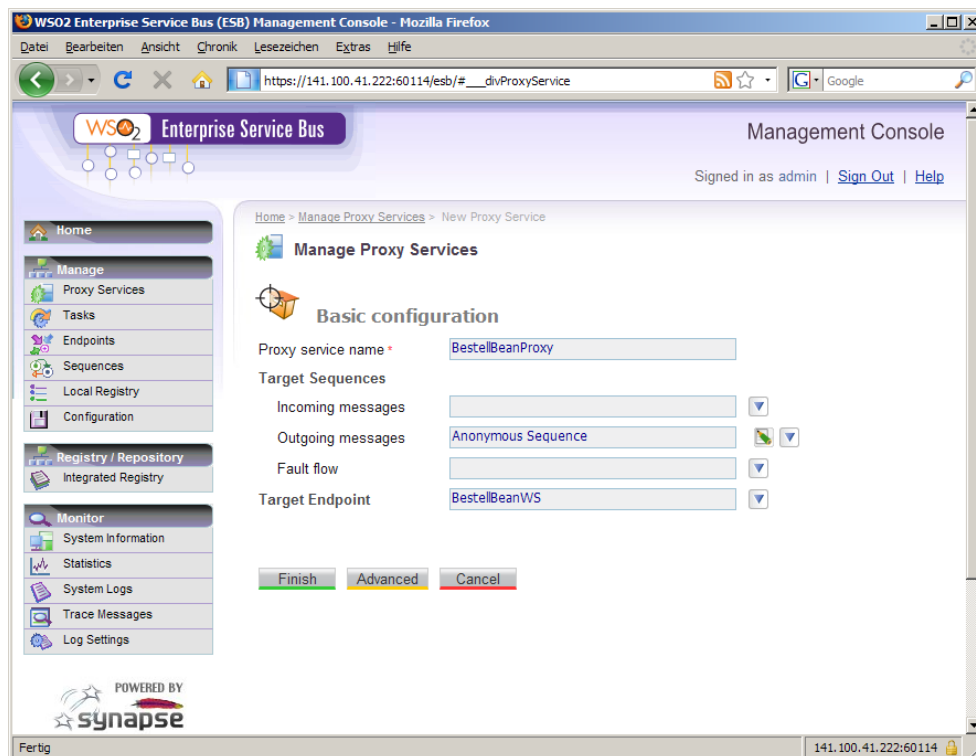


Abbildung 21: Hinzugefügte Sequence

Auf der folgenden Seite (siehe Abbildung 22) können die *Checkboxes* für MAILTO und VFS **deaktiviert** werden. Danach ist der **Next**-Button zu drücken.

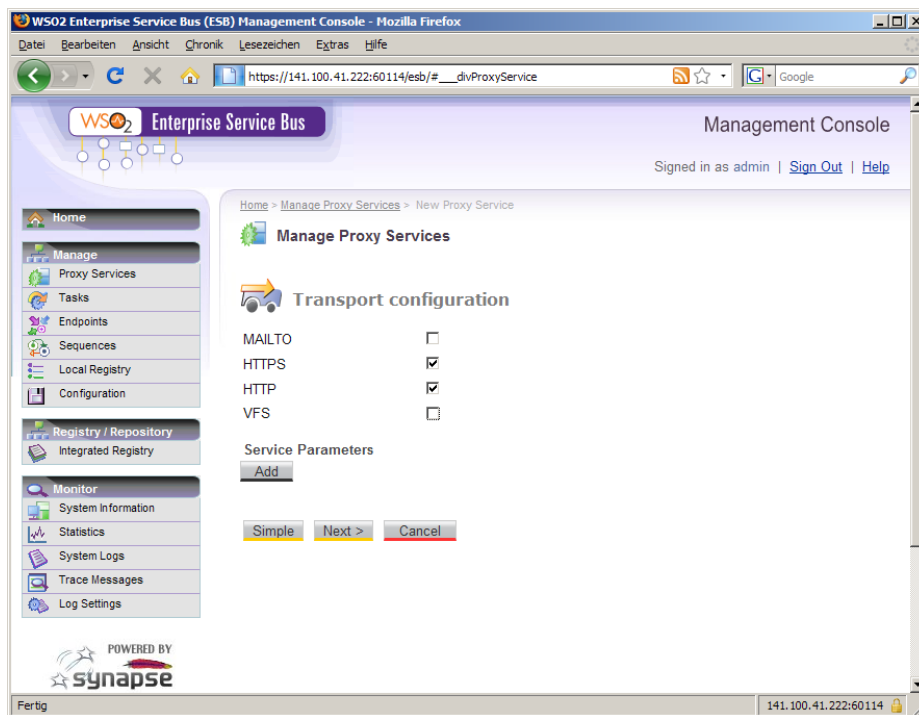


Abbildung 22: MAILTO und VFS werden nicht benötigt.

An der *QoS Configuration* (Quality of Service) sollen keine Änderungen vorgenommen werden (siehe Abbildung 23). Einfach den **Next**-Button drücken.

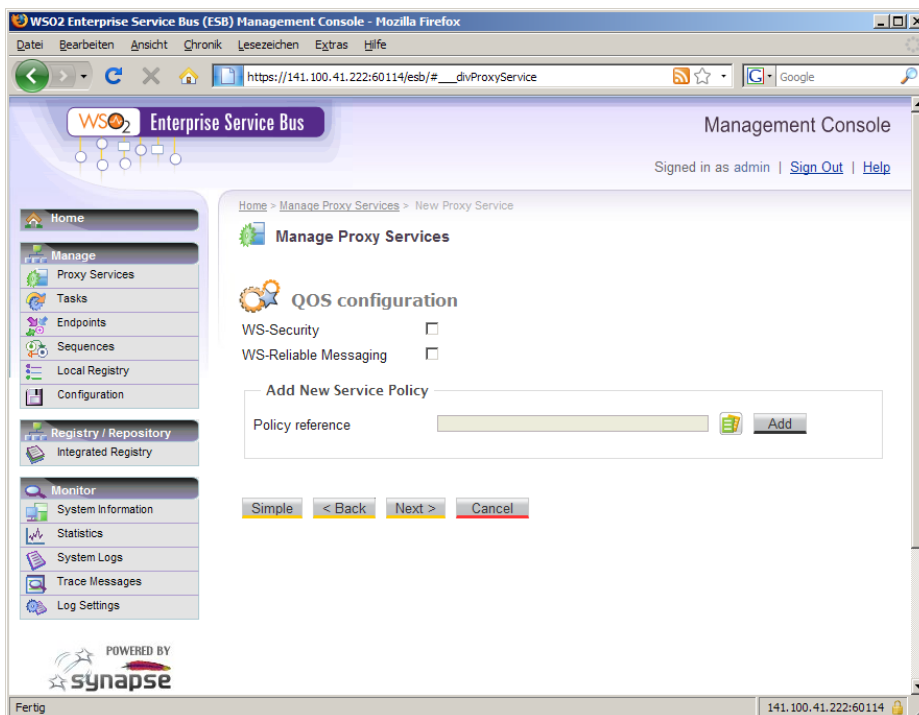


Abbildung 23: Die QoS Configuration

Bei der **Miscellaneous Configuration** soll unter **Publishing WSDL** die WSDL-Datei des aufzurufenden Web-Services angegeben werden. *WSO<sub>2</sub> ESB* liest diese WSDL-Datei ein, und generiert anhand dieser eine eigene WSDL-Beschreibung für den Proxy. Hier ist der Menue-Punkt **Specify source URL** zu wählen (siehe Abbildung 24).

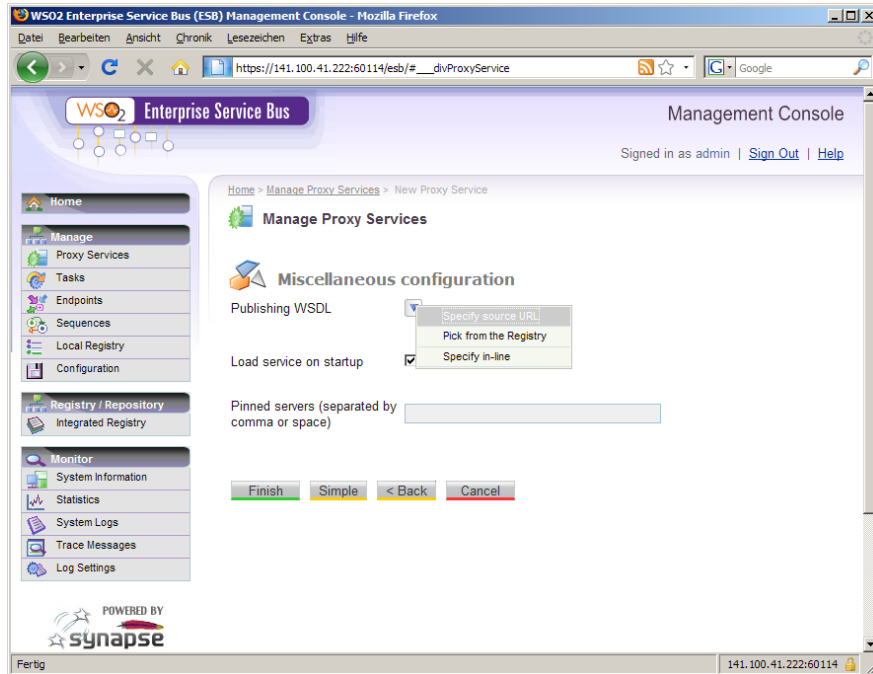


Abbildung 24: Miscellaneous Configuration

Bei **Source URL** ist die URL der WSDL-Beschreibung anzugeben (siehe Abbildung 25). Sie sollte wie folgt lauten: **<http://<ihre-ip-adresse>:8080/BestellBeanService/BestellBean?WSDL>** .

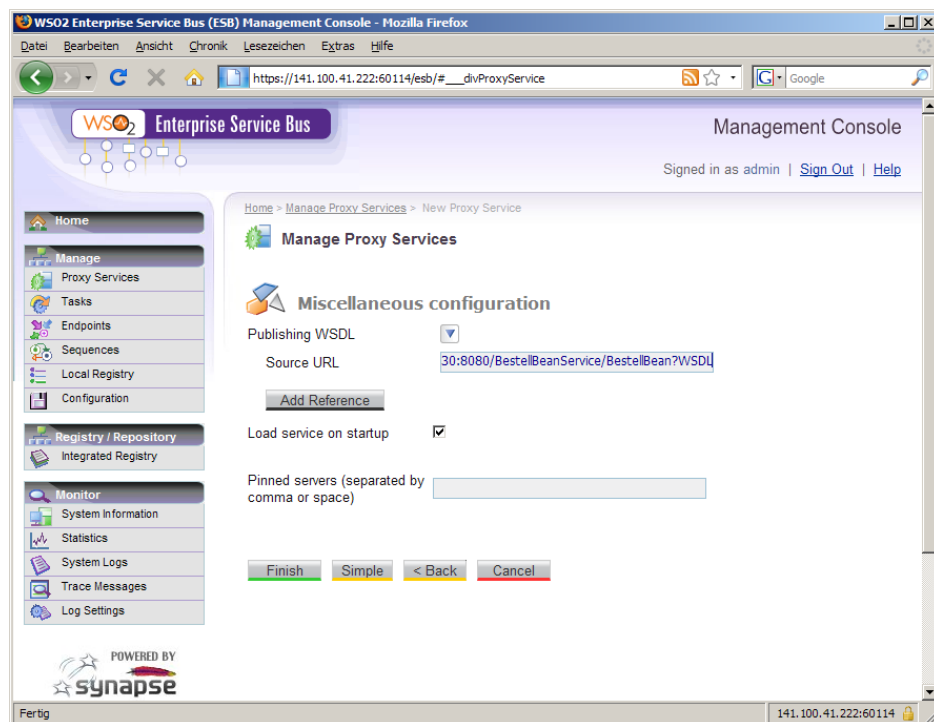


Abbildung 25: Festlegen der Publishing WSDL

Nun ist der **Finish**-Button zu drücken. Unter **Manage** → **Proxy Services** wird nun der soeben erstellte Proxy angezeigt (siehe Abbildung 26).

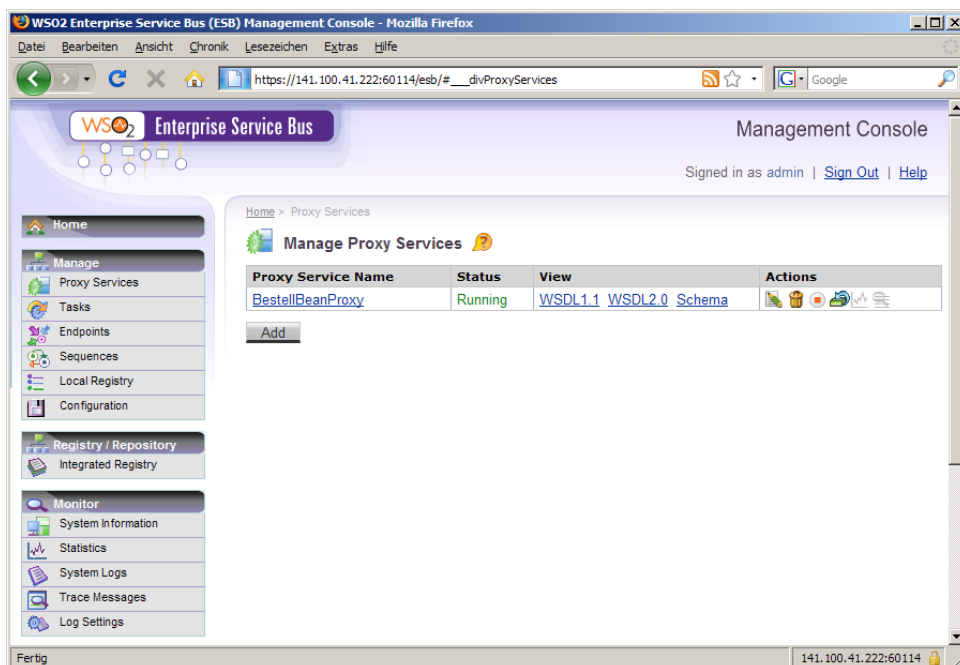


Abbildung 26: Der soeben erstellte Proxy.



Nun ist ein guter Zeitpunkt zum Speichern der Änderungen auf die Festplatte des Servers. Um dies zu tun, muss auf der Seite **Manage** → **Configuration** der **Save**-Button gedrückt werden.

### 1.4.3 Testen des Proxies aus SoapUI

Die SOAP-Engine von *WSO<sub>2</sub> ESB* (Apache Axis2) ist so konfiguriert, dass sie den Port welcher auf Seite 8 als **Soap-Port** definiert wurde verwendet.

Wenn nun im Web-Browser die URL <http://141.100.41.222:<soap-port>/soap/> aufgerufen wird, sollte dort der Eintrag **BestellBeanProxy** vorhanden sein (siehe Abbildung 27). Durch Anklicken des Links gelangt man zur WSDL-Datei des Web-Service-Proxies.

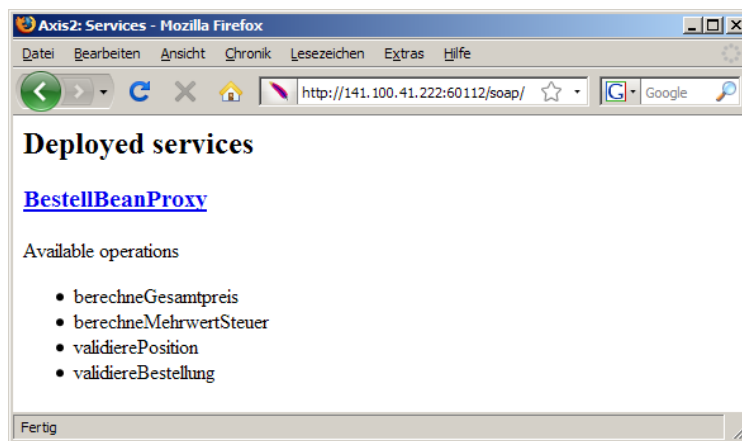


Abbildung 27: Der von WSO2 ESB bereitgestellte Web-Service-Proxy.

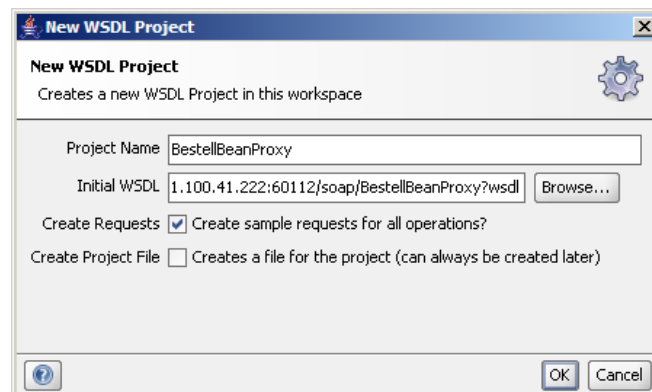


Abbildung 28: Erstellen des SoapUI-Projekts für den BestellBeanProxy.

Die URL der WSDL-Datei ist zu kopieren und analog zu Kapitel 1.1 ist in *SoapUI* ein neues WSDL-Projekt zu erstellen, welches den Namen **BestellBeanProxy** erhalten sollte (siehe Abbildung 28).

Sollte es zu einer Fehlermeldung kommen, vergewissern Sie sich, dass sie die URL der WSDL-Datei korrekt verwendet haben.

Beim neu generierten WSDL-Projekt fällt auf, dass zwei Schnittstellen generiert wurden. Eine für SOAP 1.1 und eine andere für SOAP 1.2. Wir werden nur die Schnittstelle für **SOAP 1.1** verwenden<sup>5</sup>. Testen sie nun, ob der Web-Service-Proxy wie erwartet funktioniert (siehe Abbildung 29).

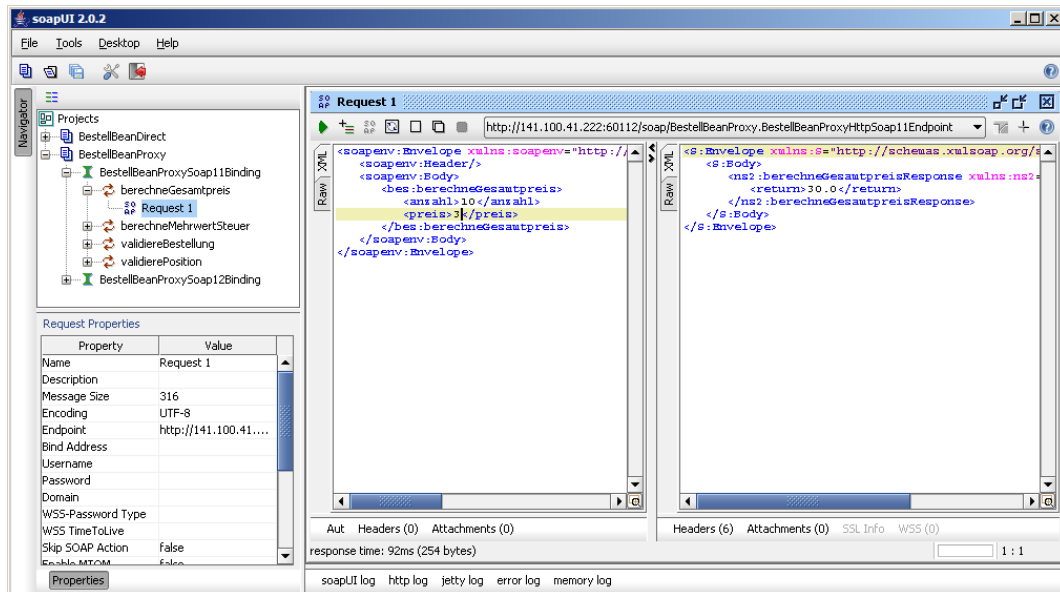


Abbildung 29: Der Proxy liefert das erwartete Ergebnis.

Beim Aufruf des Proxies wird nun vom WMS der auf dem GlassFish-Server *deployte* Web Service aufgerufen, und dessen Antwortmeldung zurückgegeben.

<sup>5</sup> Dies ist darin begründet, dass *Glassfish* mit dem bei SOAP 1.2 verwendeten MIME-Type *application/soap+xml* Probleme hat.



## 1.5 Hinzufügen der Authentifizierung

Der Proxy, der soeben erstellt wurde, stellt bis jetzt noch keinen Mehrwert dar. Der nächste Schritt soll daher das Hinzufügen eines *Mediators* zur Authentifizierung sein. Der *Mediator* hierfür ist als Java-Modul bereits implementiert.<sup>6</sup>

### 1.5.1 Der Ablauf der Mediation

Der zu verwendende Mediator hat sowohl die Aufgabe für die Authentifizierung zu sorgen, als auch die Protokollierung vorzunehmen. Der Logische Ablauf ist in folgende Phasen unterteilt.

Für die Eingehenden Nachrichten sieht der Ablauf folgendermaßen aus:

1. Die Protokollierung wird gestartet
2. Die Authentifikation wird durchgeführt
3. Dem Protokoll wird mitgeteilt, dass nun die Backend-Phase beginnt

Für die Ausgehenden Nachrichten sieht die Mediation so aus:

1. Dem Protokoll wird mitgeteilt, dass die Backend-Phase vorbei ist
2. Die Protokollierung wird beendet

Diese einzelnen Schritte verwenden jeweils einen *Class-Mediator*, welcher die Java-Klasse mit den entsprechenden Parametern aufruft. Sie wurden bereits in den Sequenzen mit den Namen EWMS... vorbereitet<sup>7</sup>.

Nun muss der **BestellBeanProxy** so bearbeitet werden, dass die entsprechenden Sequenzen aufgerufen werden.

---

<sup>6</sup> Der Mediator wurde im Rahmen einer Masterarbeit von Buchholz/Tarsia entwickelt.

<sup>7</sup> Wer die Funktionsweise der Mediatoren nachvollziehen will, kann ihren Quellcode im Verzeichnis **D:\DSI\Projekte\DSI-Praktikum-2\ausgang\WMS\_modified\EclipseWorkspace\ewMSMediators\src\tarbu** betrachten.

## 1.5.2 Bearbeiten des Proxy Services

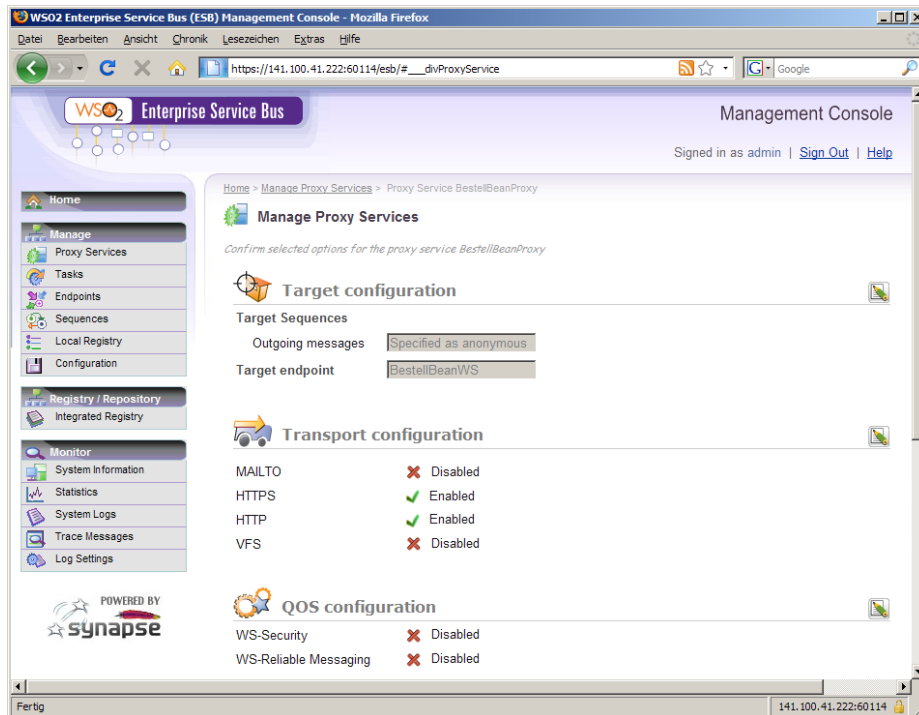



Abbildung 30: Bearbeiten des Proxy Services.

Wählen Sie in der Management-Console den Punkt **Manage** → **Proxy Services** aus. Wählen Sie dort den **BestellBeanProxy** an, oder klicken Sie auf das Bearbeiten-Symbol (  ). Drücken Sie den Edit-Button rechts von Target-Configuration (siehe Abbildung 30).

Hier müssen nun die *Target-Sequences* für die ein- und ausgehenden Nachrichten so geändert werden, dass die bereits vorbereiteten Sequenzen anhand des in Kapitel 1.5.1 beschriebenen Ablaufs aufgerufen werden.

Für die *Incoming messages* bedeutet dies, dass eine neue *Sequence* benötigt wird, welche mittels **Specify as Anonymous** erstellt werden sollte (siehe Abbildung 31).

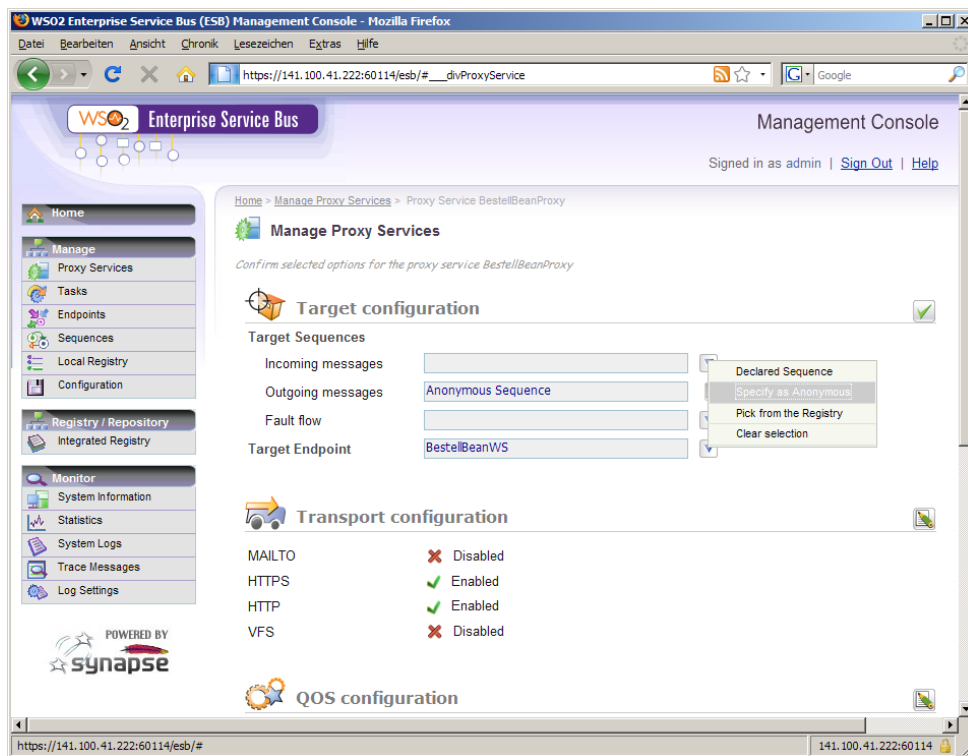



Abbildung 31: Erstellen einer neuen anonymen Sequence.

In diese *Sequence* sollten wiederum mittels drücken des **Add Mediator**-Buttons und der Auswahl des Punktes **Core** → **Sequence** im Popup-Menü hintereinander drei Sequenzen eingefügt werden. Im Sequence Mediator-Fenster ist das **Registry Browser**-Symbol (  ) anzuklicken (siehe Abbildung 32).

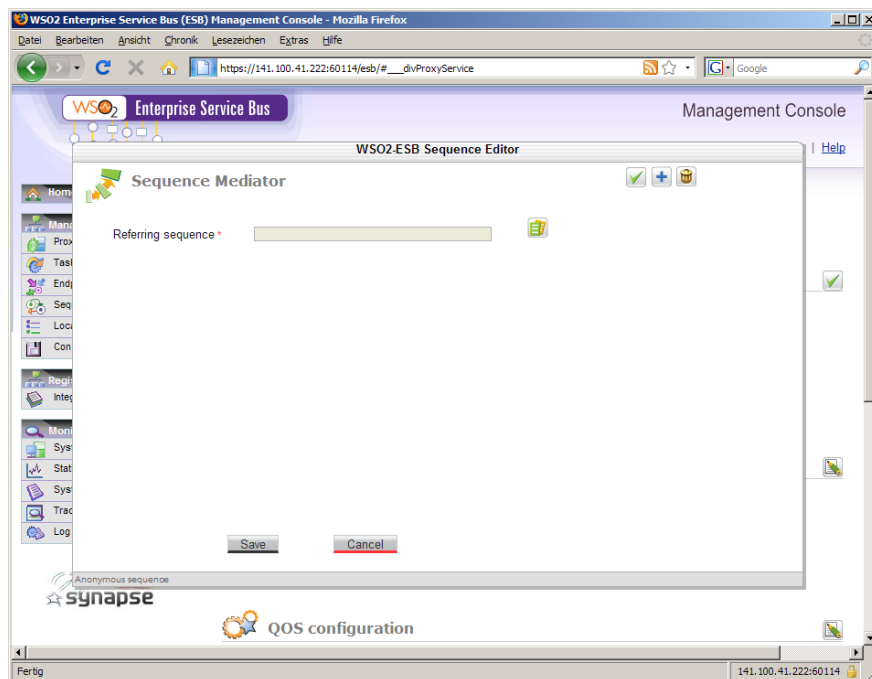





Abbildung 32: Der Sequence Mediator.

Im daraufhin erscheinenden Dialog kann unter **Local Registry** die gewünschte aufzurufende Sequenz ausgewählt werden (siehe Abbildung 33). Für die Sequenz der *Incoming messages* sind dies die Sequenzen:

1. EWMSProtocolStartMediationSequence
2. EWMSAuthenticationSequence
3. EWMSStartBackendSequence

Nach der Auswahl der Sequenz muss noch mittels dem Update-Icon (  ) die Änderung bestätigt werden. Mittels der Move Up- und Move Down-Icons (   ) kann im Nachhinein die Reihenfolge der Mediatoren geändert werden.

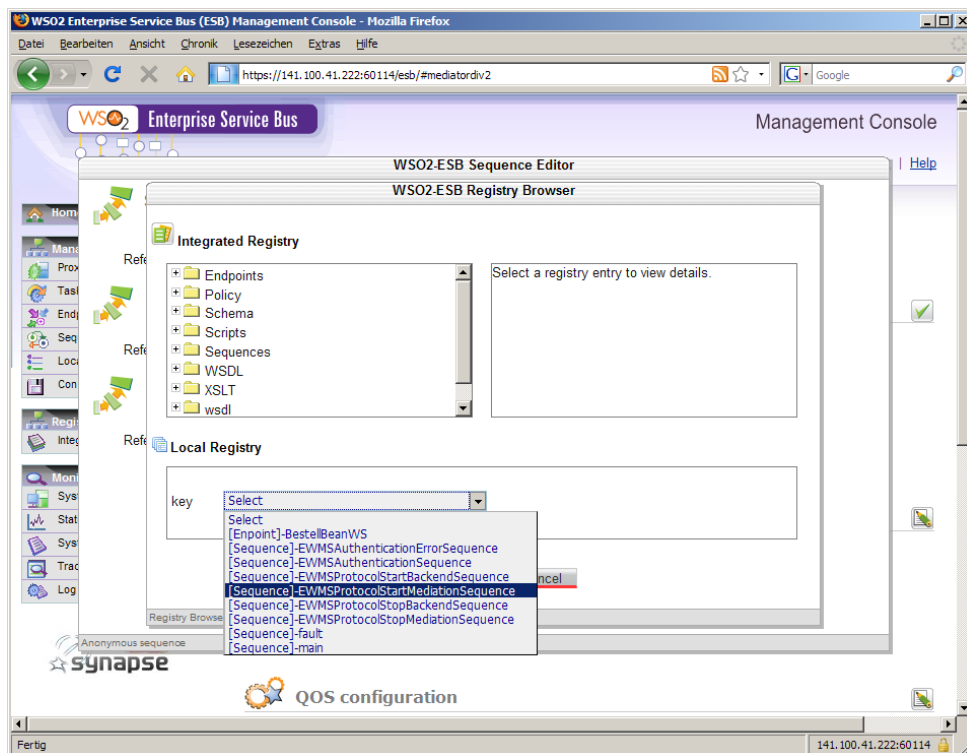


Abbildung 33: Auswahl der aufzurufenden Sequenz.

Schließlich sollte die Sequenz wie in Abbildung 34 aussehen.

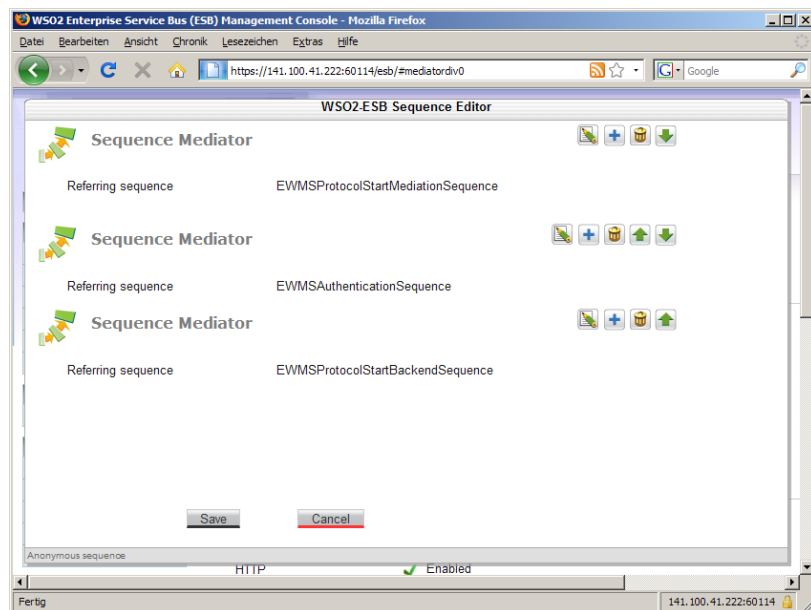


Abbildung 34: Fertiggestellte Sequenz für die eingehenden Nachrichten.

Für die *Outgoing messages* ist die selbe Methode anzuwenden. Die Sequenzen hierfür sind:

1. EWMSProtocolStopBackendSequence
2. EWMSProtocolStopMediationSequence

Danach soll der *Send-Mediator* folgen. Die Sequenz sollte nun wie in Abbildung 35 aussehen. Nach dem Bestätigen der Änderungen wäre wieder ein guter Zeitpunkt zum Aufrufen von **Manage** → **Configuration** → **Save**. Wenn jetzt aus *SoapUI* heraus der BestellBeanProxy aufgerufen wird, sollte ein AUTHENTICATION-ERROR gemeldet werden (siehe Abbildung 36).

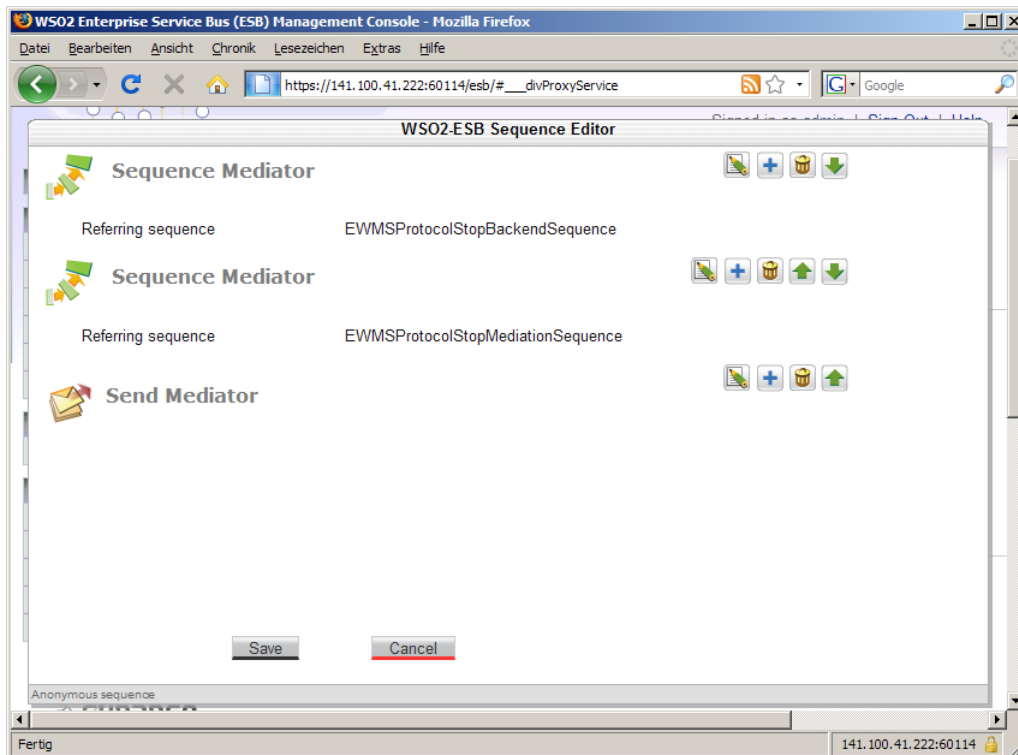


Abbildung 35: Fertiggestellte Sequenz für die ausgehenden Nachrichten.

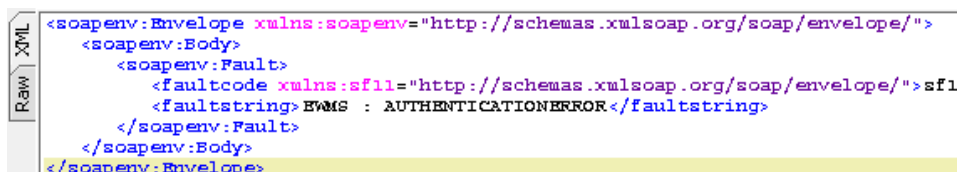


Abbildung 36: Der Proxy meldet einen Authentication-Error

## 1.6 Editieren der Datenbank-Einträge

Im letzten Schritt wurde der Authentifizierungs-Mediator in den Web-Service-Proxy eingebunden. Das heißt bei jedem Aufruf des Proxies wird nun die Authentifizierung überprüft, und der Aufruf wird mitgeloggt. Eine Frage die noch zu klären ist, ist wo der Benutzername sowie das Passwort für die Authentifizierung gespeichert werden, und wo das Protokoll abgelegt wird.

Diese Daten werden in einer *JavaDB*<sup>8</sup>-Datenbank gespeichert. Die Datenbank wird auch auf dem Server ausgeführt.

Der Zugriff auf die *JavaDB*-Datenbank ist aus *Netbeans* heraus möglich. Hierzu ist in den Services-Reiter zu wechseln, und nach einem Rechtsklick auf **Databases** der Menüpunkt **New Connection** zu wählen (siehe Abbildung 37).

Im daraufhin erscheinenden Dialog (siehe Abbildung 38) sind ist zunächst der Radio-Button **Direct URL Entry** zu selektieren. Danach sind folgende Daten einzugeben:

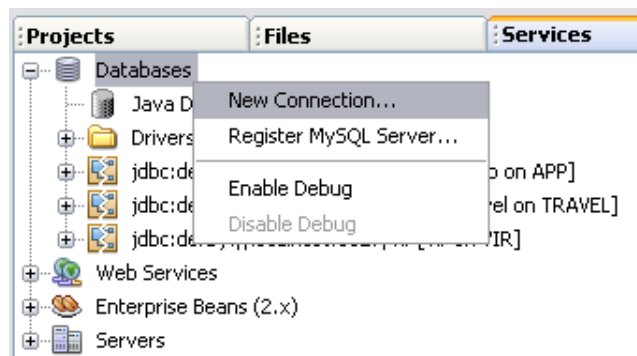


Abbildung 37: Verbindung zur Datenbank anlegen.

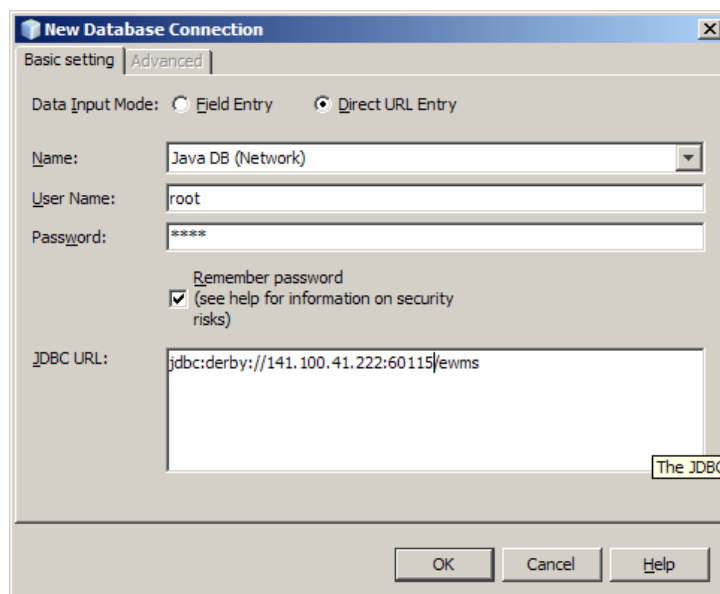


Abbildung 38: Parameter für die Verbindung

Name	Java DB (Network)
Database URL	jdbc:derby://141.100.41.222:<java-db-port>/ewms
Nutzername	root
Passwort	root

Nach der Eingabe sollte der **Ok**-Button gedrückt werden.

<sup>8</sup> Bei *JavaDB* handelt es sich um eine kleine (2.5 MB) von der Firma Sun *supportete* Datenbank, auf Basis von Apache Derby.

Im darauf folgenden Dialog soll das Schema ROOT ausgewählt werden (siehe Abbildung 39).

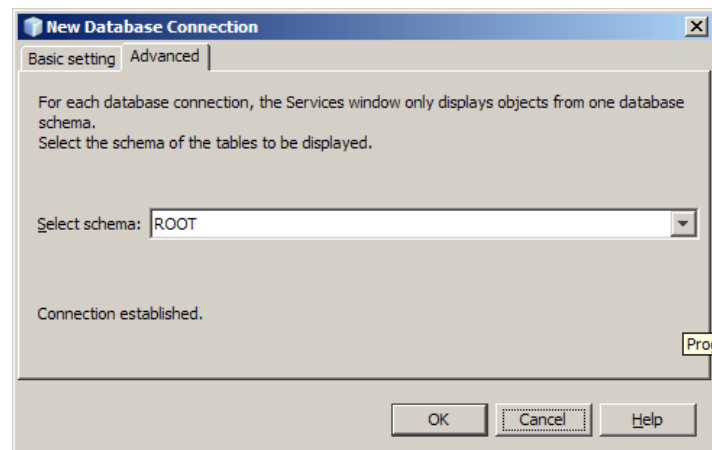


Abbildung 39: Auswahl des ROOT-Schemas.

Unter dem Knoten *Databases* wurde nun eine neue Verbindung hinzugefügt (siehe Abbildung 40).

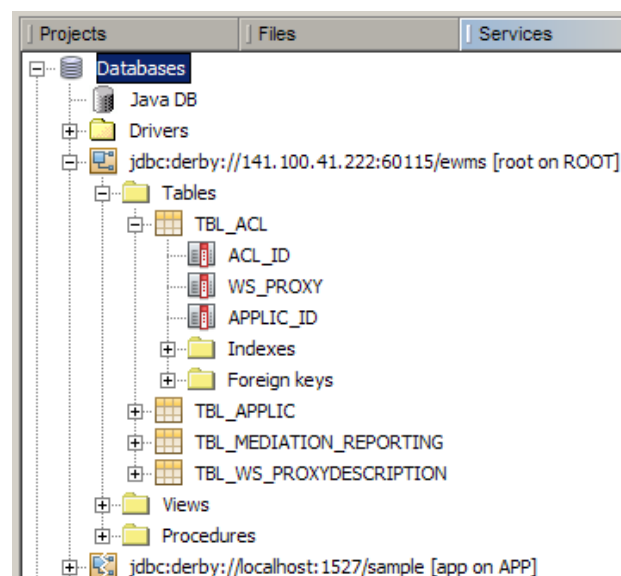



Abbildung 40: Die Verbindung zur Datenbank wurde hergestellt.

Über den Eintrag **View Data** im Kontext-Menue eines Tabellen-Knotens lassen sich die Daten einer Tabelle anzeigen (siehe Abbildung 41). Oberhalb der Tabelle kann ein SQL-Befehl eingegeben werden, der nach dem Drücken des -Buttons ausgeführt wird (siehe Abbildung 42). Seit *Netbeans* 6.5 lassen sich die Datenbank-Tabellen auch direkt über die Tabellenansicht (siehe Abbildung 42) bearbeiten.



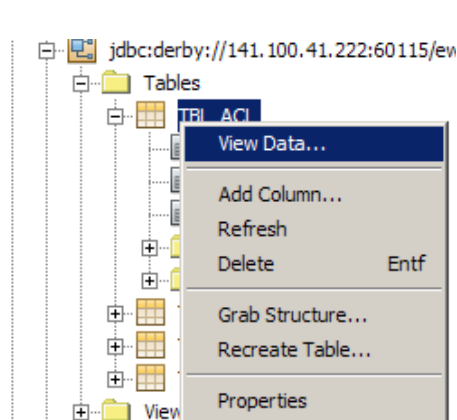


Abbildung 41: Tabellendaten anzeigen

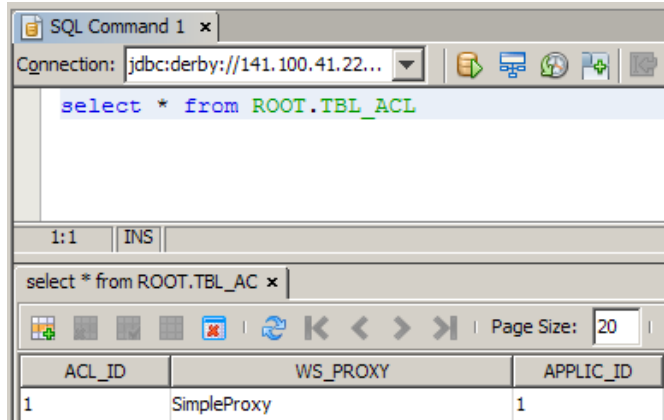


Abbildung 42: Anzeigen und Bearbeiten einer Tabelle.

Um für den BestellBeanProxy einen Namen und ein Passwort zu setzen, muss zunächst die Tabelle **tbl\_acl** (Access Control List) bearbeitet werden.

In dieser Tabelle ist eine neue Zeile für den **BestellBeanProxy** hinzuzufügen.

acl_id	2
ws_proxy	BestellBeanProxy
applic_id	2

Dieser Datensatz ordnet dem BestellBeanProxy eine **applic\_id** (=2) zu. Über die **applic\_id** wird schließlich eine Kombination aus Name und Passwort zugewiesen.

Um diese Zeile einzufügen ist folgender SQL-Befehl einzugeben und auszuführen<sup>9</sup>:

```
INSERT INTO tbl_acl (acl_id, ws_proxy, applic_id) VALUES (2, 'BestellBeanProxy', 2);
```

Um das Ergebnis des SQL-Befehls zu betrachten, muss wieder mittels **View Data** der Tabelleninhalt angezeigt werden.

In der Tabelle **tbl\_applic** wird jeder **applic\_id** eine Kombination aus Name und Passwort zugeordnet. Für unsere **applic\_id=2** muss ein neuer Eintrag hinzugefügt werden, der den Benutzernamen, sowie das Passwort enthält, durch eine SQL-Anweisung geändert werden..

```
INSERT INTO tbl_applic (applic_id, applic_name, applic_passwd) VALUES (2, 'neuer Name', 'neues Passwort');
```

Name und Passwort sollten Sie sich merken, denn sie werden bald wieder benötigt.

Für die Protokollierung wird die Tabelle **tbl\_mediation\_reporting** verwendet. Diese Tabelle ist zu diesem Zeitpunkt noch leer. Es folgt eine Erklärung deren Spalten.

<sup>9</sup> Alternativ zur Verwendung der SQL-Befehle können die Daten auch in der entsprechenden Tabelle bearbeitet werden (Übernehmen der Änderungen nicht vergessen).

mediation_id	Eindeutige Id der Mediation
applic_id	Id die der Name/Passwort-Kombination zugeordnet ist
ws_proxy	Name des Proxies
result_mediation	Ergebnis der Mediation
log_cleartext	Zusätzlicher Logging-Text
ws_proxy_mtime_totale	Zeit die für die Mediation benötigt wurde
ws_proxy_betime	Zeit die für die Backend-Phase benötigt wurde
ws_proxy_mtime_start	Zeitpunkt zu dem mit der Mediation begonnen wurde

## 1.7 Testen der Authentifizierung in SoapUI

Nachdem diese Änderungen an der Datenbank vorgenommen wurden, kann nun aus *SoapUI* heraus die Authentifizierung getestet werden. Starten Sie hierfür wieder *SoapUI*, und senden Sie eine Nachricht an den **BestellBeanProxy**.

Die Response-Nachricht sollte eine Fehlermeldung enthalten (siehe Abbildung 43).

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <soapenv:Fault>
      <faultcode xmlns:sf11="http://schemas.xmlsoap.org/soap/envelope/">sf11:s
      <faultstring>BWMS : AUTHENTICATIONERROR</faultstring>
    </soapenv:Fault>
  </soapenv:Body>
</soapenv:Envelope>
```

Abbildung 43: Das WMS liefert einen Authentifizierungs-Fehler

Damit die Anfrage an den Proxy wie gewohnt bearbeitet wird, muss in *SoapUI* im linken unteren Bereich, in den **Request Properties** der **Username** und das **Password** gesetzt werden. Beim nochmaligen Senden der Nachricht sollte nun das gewohnte Ergebnis zurückgegeben werden (siehe Abbildung 44).

Sollte es dennoch die Fehlermeldung gesendet werden, kontrollieren Sie ob in *SoapUI* unter **File** → **Preferences** in der Kategorie HTTP Settings die Checkbox **Authenticate preemptively** aktiviert ist.

## 1.8 Testen der Protokollierung in SoapUI

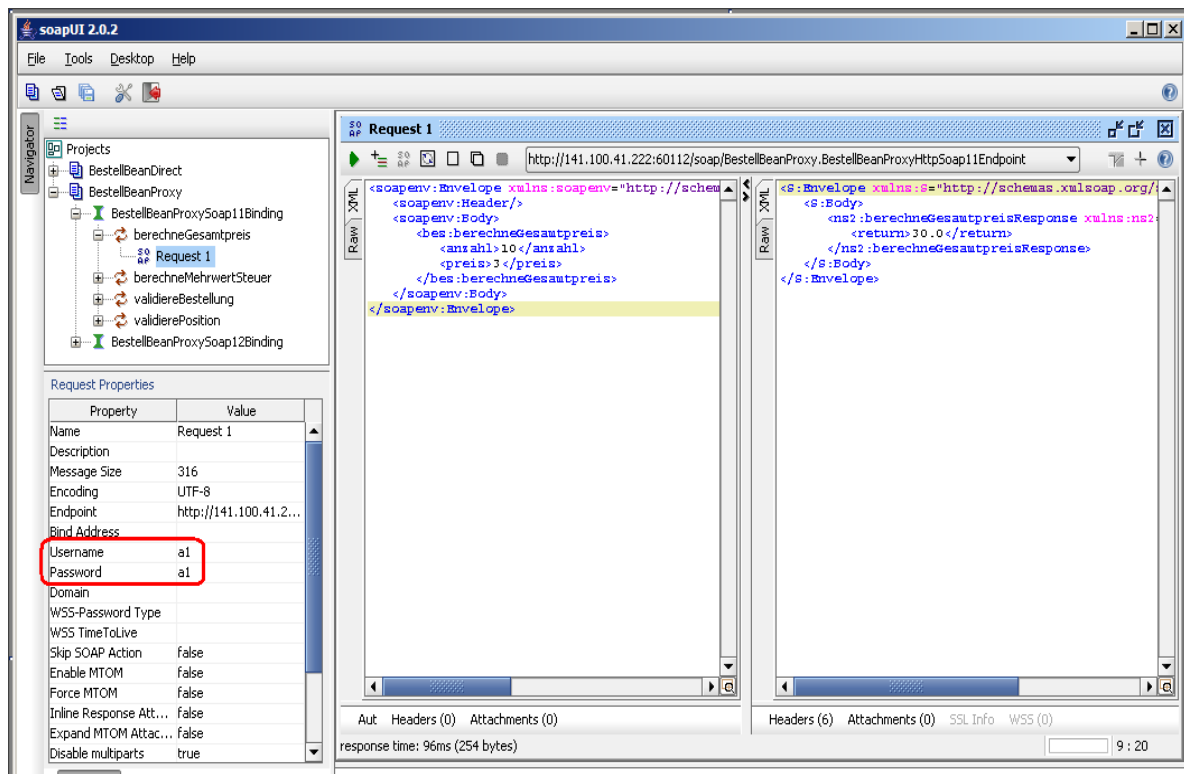


Abbildung 44: Nach dem setzen des Benutzernamens und des Passwortes antwortet der Proxy wie gewohnt.

Nun können aus *SoapUI* heraus Nachrichten an den *BestellBeanProxy* gesendet werden. Die Protokollierung der einzelnen Mediationen kann über die Verbindung zur Datenbank aus *Netbeans* heraus überprüft werden. Hierzu ist die Tabelle **tbl\_mediation\_reporting** zu betrachten. Falls die Einträge nicht in der richtigen Reihenfolge angezeigt werden, sollte folgender SQL-Befehl benutzt werden, welcher garantiert dass die neuesten Einträge oben angezeigt werden:

```
SELECT * FROM tbl_mediation_reporting t ORDER BY mediation_id DESC;
```

Die Ergebnis-Ansicht sollte etwa wie in Abbildung 45 aussehen.

Connection: jdbc:derby://141.100.41.222:60115/ewms [...]

```
select * from ROOT.TBL_MEDIATION_REPORTING t ORDER BY mediation_id DESC;
```

1:46 | INS

select \* from ROOT.TBL\_ME x

Page Size: 20 | Total Rows: 6

MEDIATION_ID	APPLIC_ID	WS_PROXY	WS_PROXY_MTIME_TOTAL	WS_PROX...	RESULT_MEDIATION
40	1	BestellBeanProxy	12	8	SUCCESS
39	-1	BestellBeanProxy	8	0	AUTHENTICATIONERROR
38	-1	BestellBeanProxy	6	0	AUTHENTICATIONERROR
37	-1	BestellBeanProxy	9	0	AUTHENTICATIONERROR
36	1	BestellBeanProxy	57	8	SUCCESS
35	-1	BestellBeanProxy	15	0	SUCCESS

Abbildung 45: Protokollierung der Mediationen (Anstelle von **APPLIC\_ID 1** müsste **APPLIC\_ID 2** stehen)

Wenn die übermittelte Name/Passwort Kombination falsch ist, wird für die **applic\_id** der Wert **-1** und bei **result\_mediation** **AUTHENTICATIONERROR** protokolliert. Wenn die Kombination korrekt ist, lauten die Werte bei **applic\_id 1** und bei **result\_mediation SUCCESS**.



Wenn Name und Passwort nicht gesendet werden ist das Ergebnis fälschlicher Weise bei **applic\_id -1** und bei **result\_mediation SUCCESS**.

## 1.9 Modifikation des WS-Client Programms

Nachdem die korrekte Funktionsweise des **BestellBeanProxy** sichergestellt wurde, soll dieser nun aus einem Java-Programm heraus angesprochen werden.

Hierzu soll die im 1. Praktikum entwickelte **WS-Client**-Anwendung so modifiziert werden, dass sie den Web-Service-Proxy mit Authentifizierung nutzt. Folgende Modifikationen sind notwendig:

1. Der Web-Service-Client für die BestellBean ist zu entfernen.
2. Ein neuer Web-Service-Client für den BestellBeanProxy ist zu erstellen.
3. Einige Methoden in der Datei Bestellung.java müssen angepasst werden, damit sie den BestellBeanProxy-Client verwenden.
4. Einige Anpassungen sind vornehmen, damit Name und Passwort gesendet werden.

Der Ausdruck Web-Service-Client wird hier wie in **Netbeans** verwendet. Gemeint sind die automatisch generierten Java-Klassen, welche den Web-Service ansprechen.

### 1.9.1 Vorbereitungen

Starten Sie zunächst noch einmal den **EJB-Client**, und erstellen Sie die Artikel- und Kunden **Demodaten** (Damit dies richtig funktioniert, muss das Enterprise-Application-Projekt mittels **Run** ausgeführt werden).

Nachdem die Datenbank gefüllt wurde, sollte noch einmal kurz sichergestellt werden, dass der WS-Client – welcher ja noch direkt den Web-Service, und nicht den Proxy benutzt – wie gewohnt funktioniert (siehe Abbildung 46).



Abbildung 46: Ausführung von WS-Client

### 1.9.2 Löschen des BestellBean Web-Service-Clients

Da **WSO<sub>2</sub> ESB** die WSDL-Datei des BestellBean-Web-Services liest und anhand dieser die WSDL-Datei für den BestellBeanProxy generiert, ist es nicht verwunderlich dass beide den gleichen *Namespace* verwenden.

Das wird problematisch sobald man in **Netbeans** die Web-Service-Clients generiert, da für beide Web-Services – Original und Proxy – gleichnamige Klassen im gleichen *Namespace* – also im selben Verzeichnis – erzeugt werden.

Um das zu verhindern wird kurzerhand der nicht mehr benötigte Web-Service-Client für den BestellBean-Web-Service gelöscht.

Dies lässt sich erledigen indem man den **Web Service References**-Knoten öffnet, und im Kontext-Menue von **BestellBean** den Punkt **Delete** auswählt (siehe Abbildung 47). Im daraufhin erscheinenden Bestätigungs-Dialog muss **Yes** gewählt werden (siehe Abbildung 48).

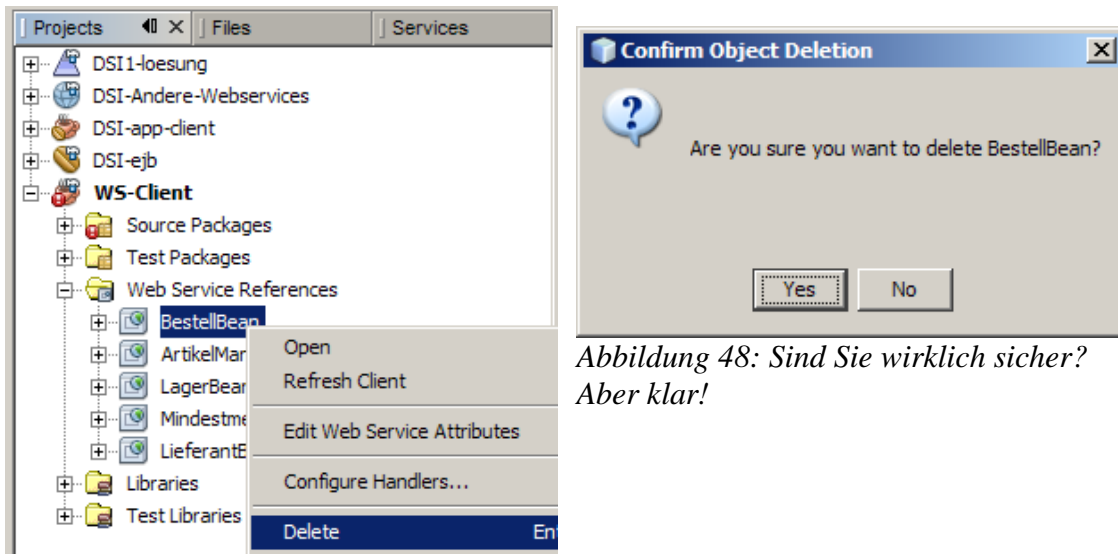


Abbildung 47: Entfernen des BestellBean-Web-Services

Abbildung 48: Sind Sie wirklich sicher?  
Aber klar!

Netbeans ist ein bisschen schwerfällig beim Erkennen von Änderungen. Wenn aber der WS-Client mit **Clean and Build** neu kompiliert wird, werden entsprechende Fehlermeldungen angezeigt (siehe Abbildung 49).

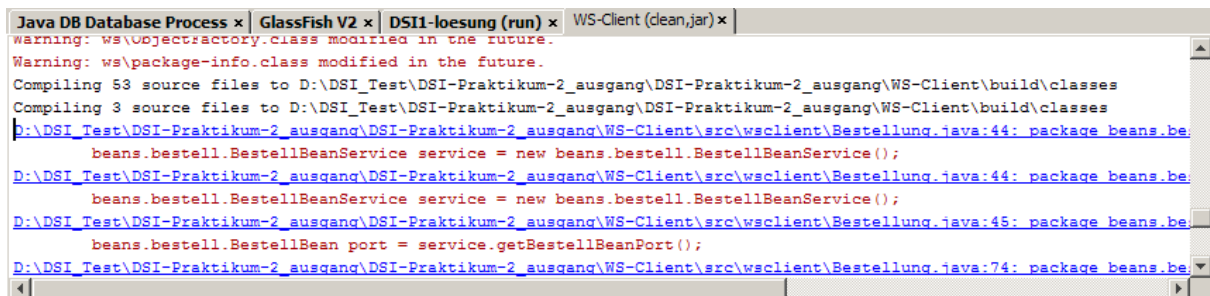


Abbildung 49: Die fehlenden Packages.

### 1.9.3 Erstellen des BestellBeanProxy-Web-Service-Clients

Nachdem der alte Web-Service-Client entfernt wurde, muss er durch einen neuen ersetzt werden, welcher den BestellBeanProxy-Web-Service nutzt.

Hierzu wird im Kontext-Menue von **WS-Client** der Punkt **New** → **Web Service Client** ausgewählt. Im Dialog-Fenster muss bei **WSDL-URL** die URL der WSDL-Datei des Web-Service-Proxies angegeben werden (<http://141.100.41.222:<soap-port>/soap/BestellBeanProxy?wsdl>). (siehe Abbildung 50) Anschließend ist der **Finish**-Button zu drücken.

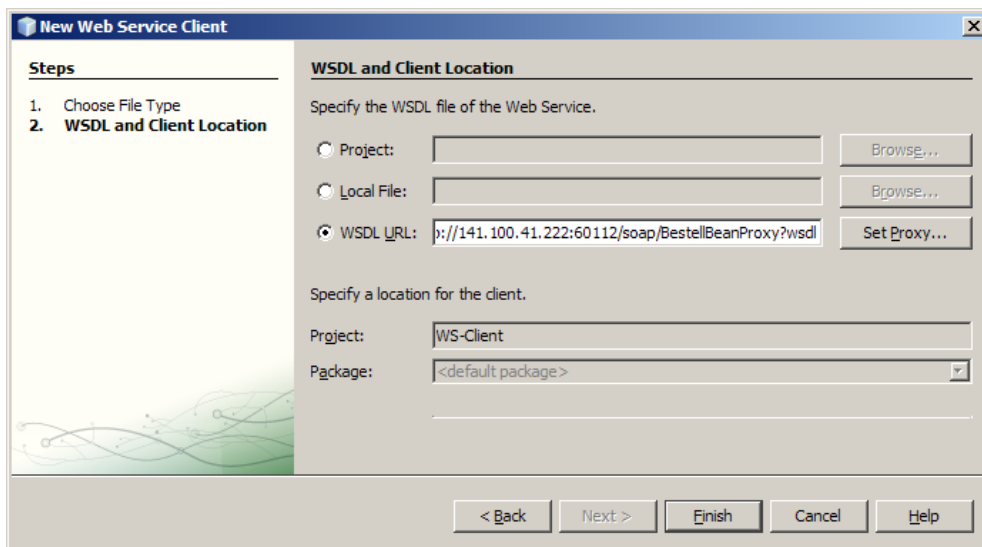


Abbildung 50: Erstellen des Web Service Clients für den Web-Service-Proxy

Beim Generieren werden zwar einige Warnungen angezeigt, ansonsten verläuft es fehlerfrei. (Netbeans 6.5: Die Fehlerdialoge können ignoriert werden)

#### 1.9.4 Anpassung der Methoden in Bestellung.java

In der Datei **Bestellung.java** müssen nun die Methoden, welche den **BestellBean**-Service nutzen, so angepasst werden, dass stattdessen der **BestellBeanProxy** genutzt wird.

Da diese Methoden - **validiereBestellung**, **berechneGesamtpreis** und **berechneMehrwertSteuer** - nur den Web-Service-Aufruf *wrappen* ist es sinnvoll ihren Inhalt zu löschen und neu zu generieren.

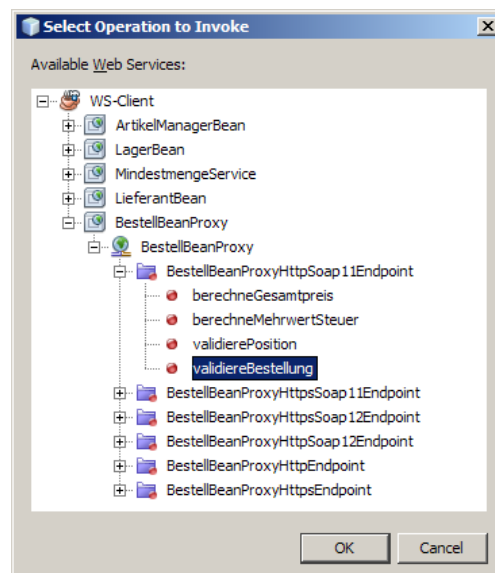


Abbildung 51: Auswahl des richtigen Web-Service Ports

Hierzu muss in jeder dieser Methoden der Web-Service-Aufruf mittels des Öffnens des Kontext-Menues im Quellcode-Editor und der Auswahl des Punktes **Web Service Client Resources** → **Call**

**Web Service Operation ...** neu generiert werden. Im Dialog für die Auswahl der aufzurufenden Operation ist hierbei jeweils der **HttpSoap11Endpoint** zu wählen (siehe Abbildung 51).

Wie im ersten Praktikum kann die try/catch-Umgebung gelöscht werden. Eventuell auftretende *Exceptions* werden bereits von den Methoden geworfen.

### 1.9.5 Setzen der Authentifizierungs-Daten

Bis jetzt wurde das Programm so modifiziert, dass der *BestellBeanProxy-Web-Service* angesprochen wird. Wenn es jetzt ausgeführt wird, wird ein Fehlerdialog angezeigt (siehe Abbildung 52).

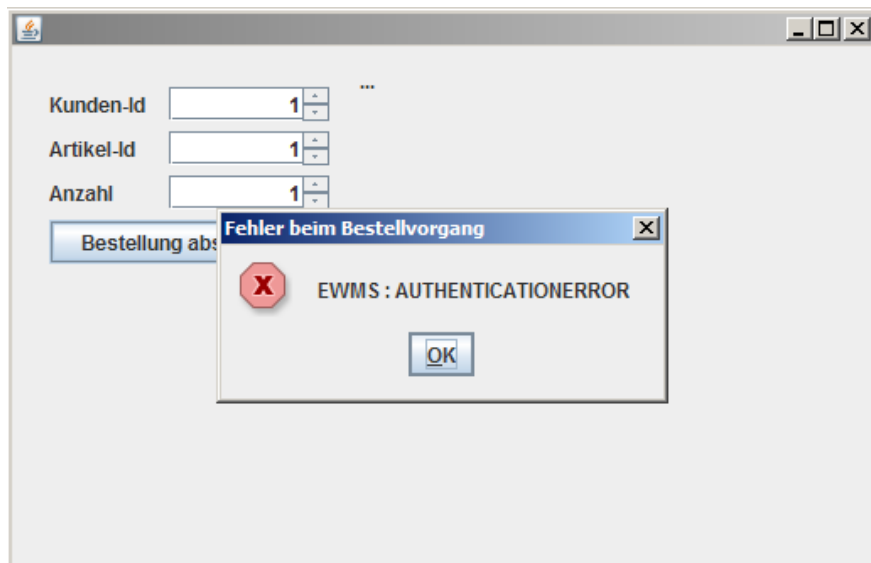


Abbildung 52: Der Fehlerdialog

Damit das Programm jedoch auf diesen Web-Service zugreifen darf, müssen noch Name und Passwort übermittelt werden.

Die folgenden Zeilen Quellcode müssen jeweils **direkt vor** dem Aufruf der jeweiligen Web-Service-Operation des *BestellBeanProxies* eingefügt werden. Das bedeutet vor die `return port.operation(parameter)`-Anweisung. Sie setzen den Benutzernamen sowie das Passwort für die HTTP-Authentication.

```
WSBindingProvider bp = (WSBindingProvider) port;
//HTTP-AUTHENTICATION
bp.getRequestContext().put(BindingProvider.USERNAME_PROPERTY, name);
bp.getRequestContext().put(BindingProvider.PASSWORD_PROPERTY, password);
```

Damit nicht zu viel Code dupliziert wird, empfiehlt sich das Auslagern in eine Methode.

```
private void setNamePassword(Object port, String name, String password) {
    WSBindingProvider bp = (WSBindingProvider) port;
    bp.getRequestContext().put(BindingProvider.USERNAME_PROPERTY, name);
    bp.getRequestContext().put(BindingProvider.PASSWORD_PROPERTY, password);
}
```

Der Aufruf dieser Methode muss folgendermaßen vor der Ausführung des Web-Service-Aufrufs platziert werden



```
setNamePassword(port, "a1", "a1");
```

Zusätzlich dazu müssen im Datei-Kopf noch die benötigten Imports eingefügt werden.

```
import com.sun.xml.ws.api.message.Headers;  
import com.sun.xml.ws.developer.*;  
import javax.xml.namespace.QName;  
import javax.xml.ws.BindingProvider;
```

### 1.9.6 Ausführung von WS-Client

Wenn nun das Programm übersetzt und ausgeführt wird, sollte es wie bisher funktionieren. Wird jedoch das falsche Passwort gesetzt, wird eine *Exception* geworfen, die in einem Fehlerdialog angezeigt wird.



Abbildung 53: Bei korrekt gesetzten Passwort: normale Ausführung.

Sofern Sie beim Ausführen der WS-Client-Anwendung einen Fehler erhalten, sollten Sie prüfen, ob das Passwort korrekt gesetzt wird und die Demodaten in der Tabelle vorhanden sind (d.h. evtl. den EJB-Client neu ausführen) (siehe Abbildung 52).

## 2 Abgabe

Pro Gruppe muss abgegeben werden:

- XML-Code aus *WSO2 ESB*
  1. Loggen Sie sich in die Management-Console ein.
  2. Wählen Sie den Punkt Manage → Configuration aus.
  3. Kopieren Sie den XML-Code aus dem Textfeld in eine Datei.
- Screenshots *SoapUI*, die eine gelungenen und einer misslungen Authentifizierung zeigen.

Bitte mailen Sie die entsprechenden Dateien unter Angabe der Gruppenmitglieder an [f.buehler@fbi.h-da.de](mailto:f.buehler@fbi.h-da.de) und [g.turetschek@fbi.h-da.de](mailto:g.turetschek@fbi.h-da.de).