



Übungen zur BPEL Schulung

Version 1.5

Kaveh Keshavarzi, Thomas Bayer, Marco Hippler, Stefan Maibücher

predic8 GmbH
Moltkestr. 40
53173 Bonn

Tel.: 0228/555 25 76-0
info@predic8.de

© 2008-2009 by predic8 GmbH, <http://www.predic8.de/>

Inhaltsverzeichnis

1. Erste Schritte.....	3
2. Testen des erstellten Prozesses	9
3. Debuggen des erstellten Prozesses.....	12
4. Compound Services	13
5. Der OrderProcess	27
5.1. Vorbereitung	27
6. Einbinden eines bestehenden Web Service	37
6.1. Vorbereitung	37
7. Compensation	51
8. Correlation.....	53
9. Event Handling	55
10. Pick	61
11. Business Activity Monitoring	66
12. Troubleshooting	68
12.1. Glassfish Probleme	68
12.2. Deployment Probleme	68

Lizenz

Dies ist das Übungsskript zu unserer [BPEL Schulung](#). Außer dem Übungsskript gibt es noch einen [Foliensatz](#). Sie dürfen das Skript zur eigenen Fortbildung oder für Fremde verwenden und diese Datei beliebig oft kopieren und verteilen. Die Voraussetzung für die oben genannten Rechte ist, dass diese Datei in unveränderter Form benutzt und weitergegeben wird.

1. Erste Schritte

Übung: Availability Process erstellen

1. Erstellen Sie unter *File / New Project* in der SOA Kategorie eine neue Composite Application *SlaughterhouseApp* .
2. Erstellen Sie unter *File / New Project* in der SOA Kategorie ein neues BPEL Module namens *SlaughterhouseProcess* .
3. Erstellen Sie im Projekt *SlaughterhouseProcess* einen neuen BPEL Prozess *AvailabilityProcess*, indem Sie mit rechts auf das Projekt klicken und hier *New / BPEL Process...* auswählen.
4. Ändern Sie den *Target Namespace* in:

`http://predic8.com/AvailabilityProcess`

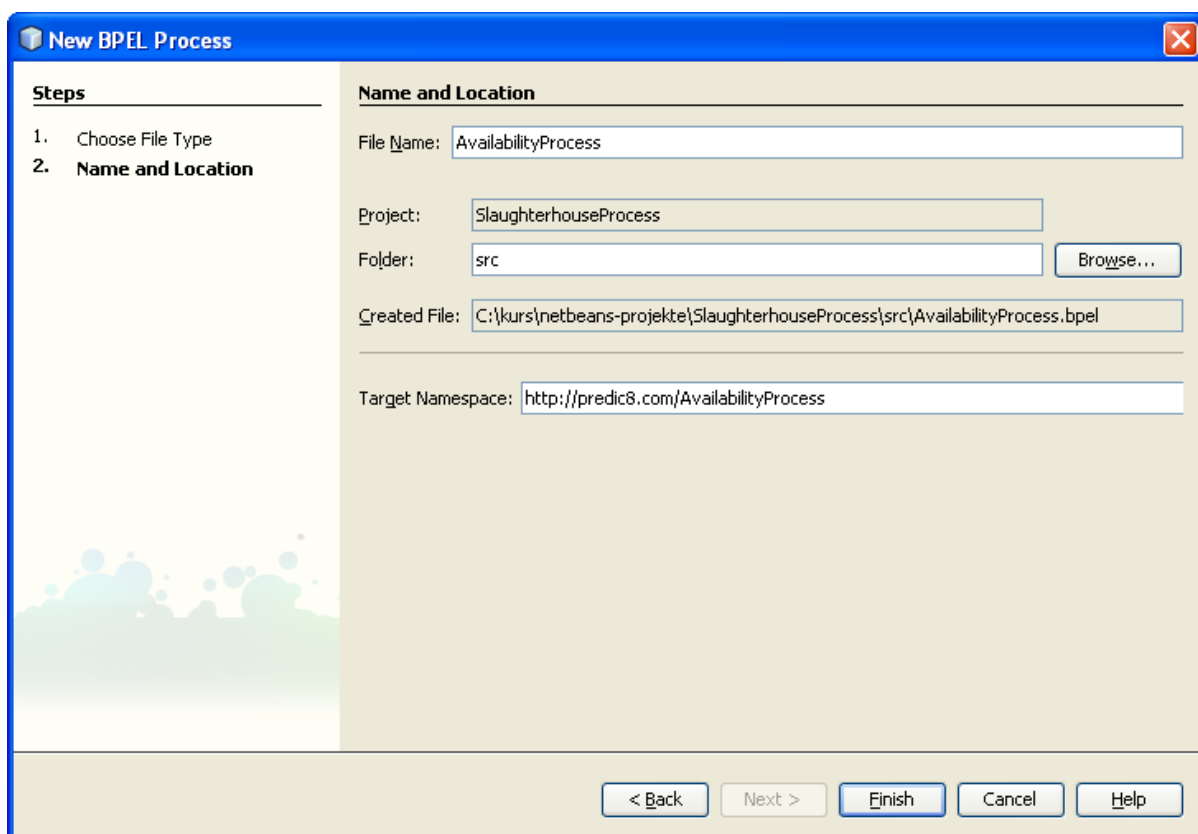


Abbildung 1: Konfiguration des BPEL Prozesses

5. Erstellen Sie im *SlaughterhouseProcess* ein neues WSDL Dokument namens *availability*, indem Sie nach einem Rechtsklick auf *SlaughterhouseProcess* auf *New / WSDL Document* klicken.
6. Ändern Sie den Target Namespace in:

<http://predic8.com/AvailabilityProcess>

New WSDL Document

Steps

1. Choose File Type
2. **Name and Location**
3. Abstract Configuration
4. Concrete Configuration

Name and Location

File Name:

Project:

Folder:

Created File:

Target Namespace:

WSDL Type:

☐ Abstract WSDL Document

☒ Concrete WSDL Document

Binding:

Type:

< Back Next > Finish Cancel Help

7. Nennen Sie den Porttype *availabilityPT*, die Operation *check* und definieren Sie zwei Inputvariablen *article* und *quantity*, sowie eine Outputvariable *availability* wie in Abbildung 2. Verwenden Sie dieselben Typen wie im Screenshot.

New WSDL Document

Steps

1. Choose File Type
2. Name and Location
3. **Abstract Configuration**
4. Concrete Configuration

Abstract Configuration

Port Type Name:

Operation Name:

Operation Type:

Input:

Message Part Name	Element Or Type
article	xsd:long
quantity	xsd:int

Add Remove

Output:

Message Part Name	Element Or Type
availability	xsd:boolean

Add Remove

Fault:

Message Part Name	Element Or Type
-------------------	-----------------

Add Remove

☒ Generate partnerlinktype automatically.

< Back Next > Finish Cancel Help

Abbildung 2: Konfiguration des WSDL Dokuments

8. Im vierten Schritt behalten Sie die Einstellungen und klicken Sie auf OK.

Übung: Editieren des BPEL Prozesses

1. Erstellen Sie im BPEL Process einen neuen PartnerLink namens *Client*.
2. Fügen Sie dem BPEL Process eine *Receive* Aktivität namens *AvailabilityRequest* hinzu und erzeugen Sie hier eine neue Inputvariable *CheckIn*, indem Sie auf *Create* klicken und hier die Einstellungen wie in Abbildung 3 vornehmen.

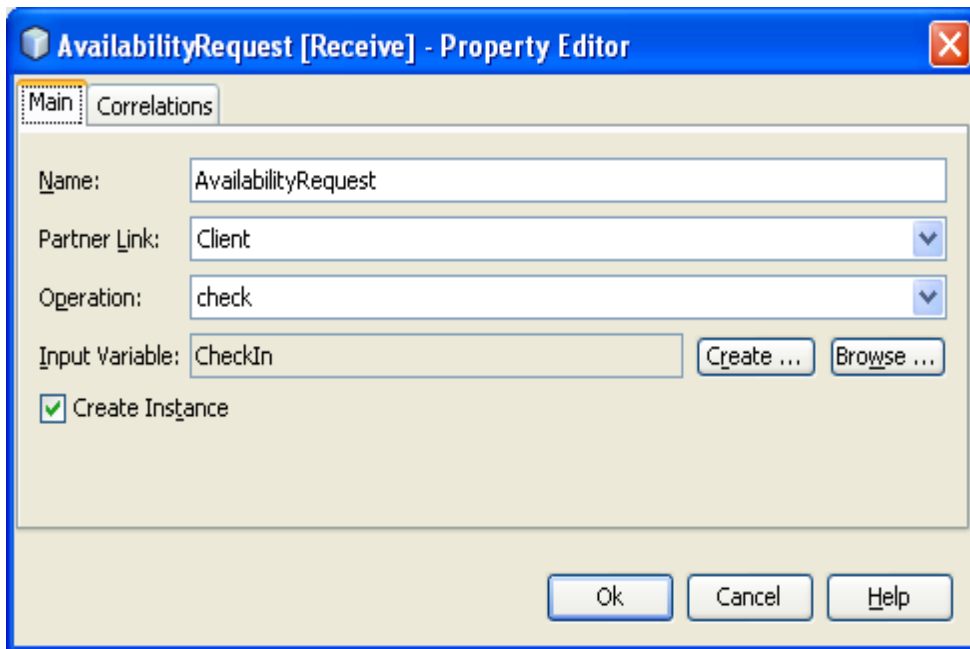


Abbildung 3: Konfiguration des *AvailabilityRequest*

3. Fügen Sie nun unter dem *AvailabilityRequest* ein Reply namens *AvailabilityReply* ein. Erzeugen Sie hier analog zum vorigen Schritt eine Output Variable *CheckOut*.

AvailabilityReply [Reply] - Property Editor

Main Correlations

Name: AvailabilityReply

Partner Link: Client

Operation: check

☒ Normal Response

Output Variable: CheckOut Create ... Browse ...

☐ Fault Response

Fault Name: Choose ...

Fault Variable: Create ... Browse ...

Ok Cancel Help

Abbildung 4: Konfiguration des *AvailabilityReply*

4. Wenn alles richtig gemacht wurde, dann sollte der Prozess nun wie in Abbildung 5 aussehen.

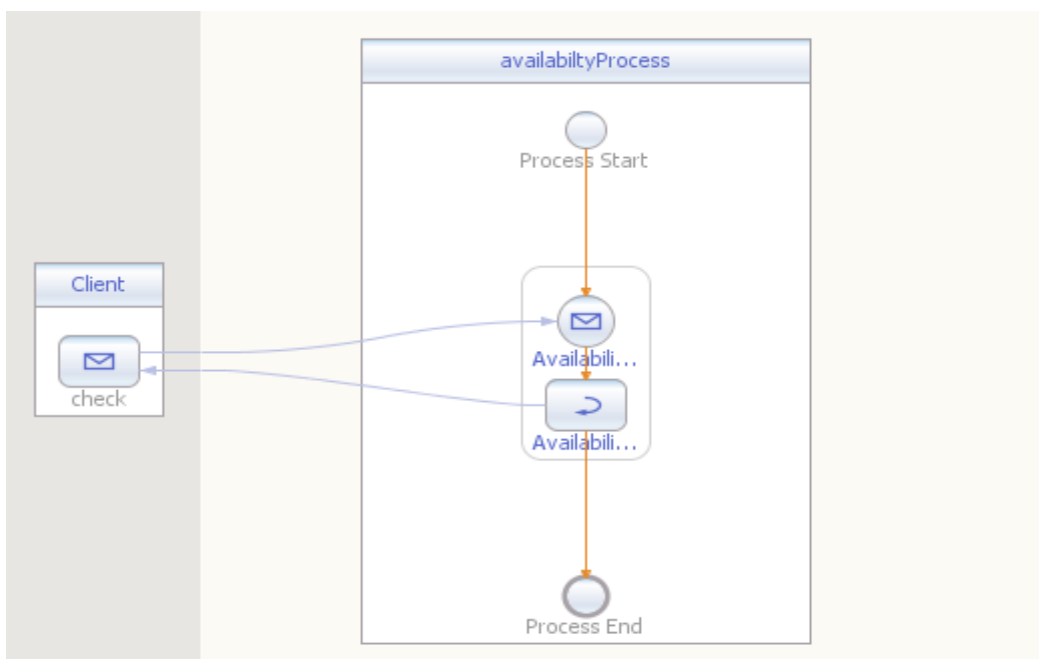


Abbildung 5: Der BPEL Prozess

Hinweis: In Netbeans 6.5 erhalten Sie eine Warning, da die *CheckIn* Variable nicht benutzt wird. Diese können Sie ignorieren.

5. Fügen Sie zwischen *AvailabilityRequest* und *AvailabilityReply* ein Assign ein, und nennen Sie es *SetAvailabilityTrue*.
6. Wählen Sie den Mapper von *SetAvailabilityTrue* und expandieren Sie die *CheckOut* Variable. Wählen Sie jetzt *availability*. Wenn Sie nun oben in der Mapperleiste das *Boolean / Logical True* wählen erscheint es in der grafischen Ansicht auf Höhe der *availability* Variable. Verbinden Sie beide mit dem orangenen Pfeil.

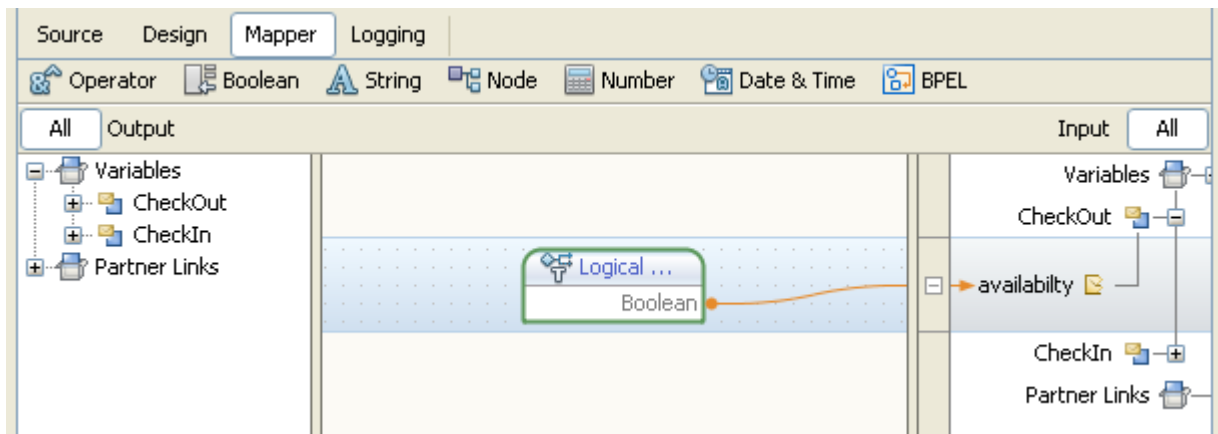


Abbildung 6: Der Mapper von *SetAvailabilityTrue*

2. Testen des erstellten Prozesses

Übung: Erstellen des Test-Cases

1. Klicken Sie mit der rechten Maustaste auf *SlaughterhouseApp* und wählen Sie „Add JBI Module“ aus. Klicken Sie nun in der linken Spalte einmal auf Ihren *SlaughterhouseProcess* und schließen Sie den Vorgang mit „Add Projekt JAR Files“ ab.
2. Erstellen Sie nun einen neuen *Test-Case*, indem Sie rechts auf das *Test-Verzeichnis* in der *SlaughterhouseApp* klicken und hier „New Test Case“ auswählen.
3. Nennen Sie Ihren Test *AvailabilityTest*.
4. Wählen Sie im nächsten Schritt die *availability.wsdl* Ihres *SlaughterhouseProcess* (Siehe Abbildung 7).

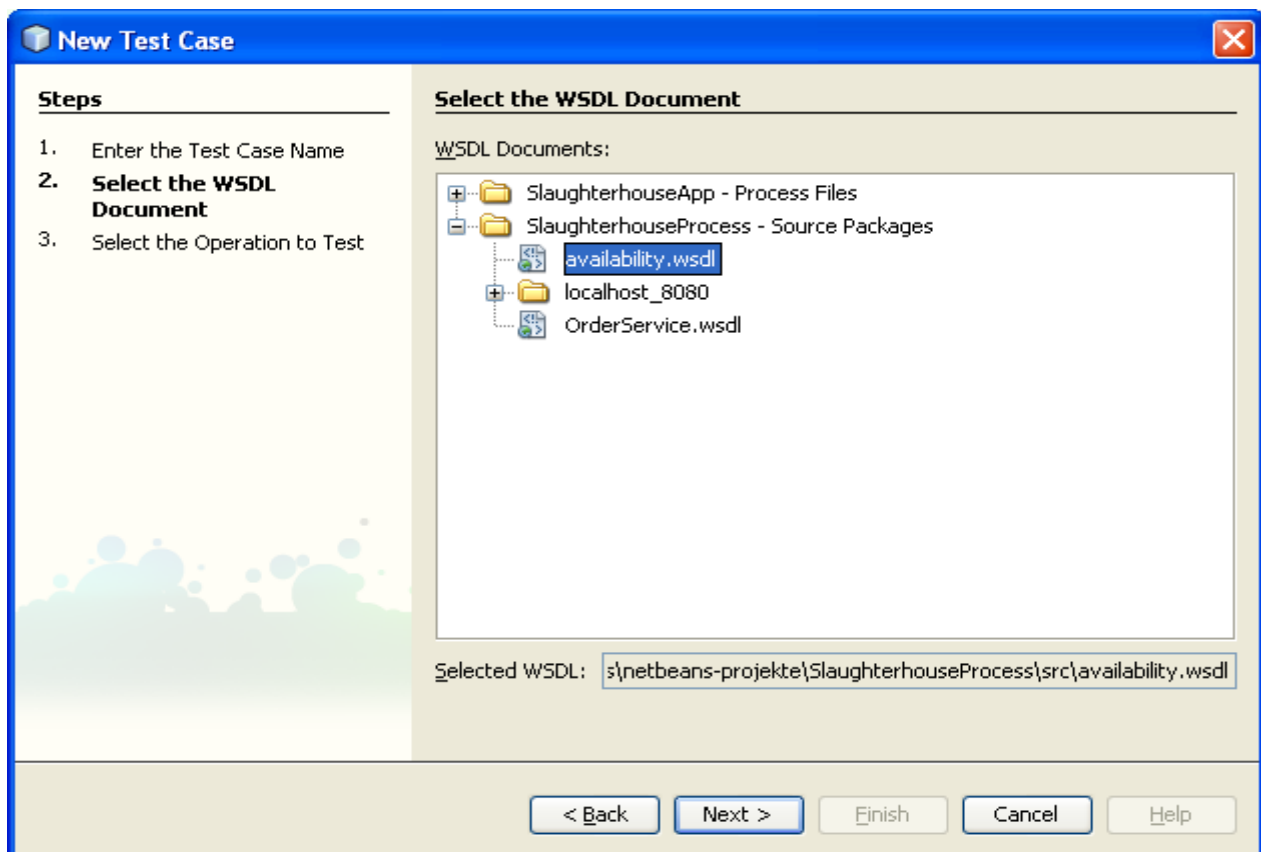


Abbildung 7: WSDL-Datei vom SlaughterhouseProcess wählen

5. Im letztem Schritt müssen Sie die *check* Operation auswählen.
6. Netbeans öffnet Ihnen daraufhin die *Input.xml* des *AvailabilityTest*. Setzen Sie hier das *article* Element auf „1“ sowie das *quantity* Element auf „100“ (jeweils ohne Anführungszeichen).

```
<soapenv:Body>
  <ava:check>
    <article>1</article>
    <quantity>100</quantity>
  </ava:check>
</soapenv:Body>
```

Übung: Ausführen des Test-Cases

1. Klicken Sie rechts auf Ihren *SlaughterhouseApp* und führen Sie *deploy* aus.
2. Nun können Sie mit den *AvailabilityTest* durch einen Rechtsklick und *Run* ausführen.
3. Die *output.xml* Datei sollte nun richtigerweise die Output-Variable *availability* auf „true“ gesetzt haben.

3. Debuggen des erstellten Prozesses

Übung: Breakpoint im BPEL-Prozess setzen

1. Öffnen Sie nun wieder *AvailabilityProcess*.
2. Klicken Sie einmal rechts auf *AvailabilityRequest* (das *Receive*-Objekt) und fügen Sie mit „*Toggle Breakpoint*“ einen Breakpoint in den Prozess hinzu.

Übung: Debuggen des Test-Cases

1. Durch einen Rechtsklick und *Debug* auf den eben erstellten *AvailabilityTest* gelangen Sie nun in den DEBUG-Modus.
2. Durch das Klicken auf den „*Step Into*“-Button bzw. das Drücken der „F7“-Taste können Sie nun Schritt für Schritt den Ablauf des Prozesses verfolgen.

4. Compound Services

Übung: PartnerLink hinzufügen

1. Öffnen Sie das Projekt SlaughterhouseService in NetBeans.
2. Betrachten Sie den Webservice *StockService* unter *Web Services*.
3. Führen Sie ein Deployment des Services durch und lassen Sie sich das WSDL Dokument anzeigen.
4. Kopieren Sie den Link des WSDL Dokuments in die Zwischenablage.
5. Fügen Sie dem *SlaughterhouseProcess* ein externes WSDL-Dokument hinzu, indem Sie mit einem Rechtsklick auf ihn mittels *New / Other / XML / „External WSDL Document(s)…”* in das entsprechende Menü gelangen.
6. Fügen Sie nun bei „From URL“ die eben kopierte Adresse des WSDL-Dokuments hinzu.

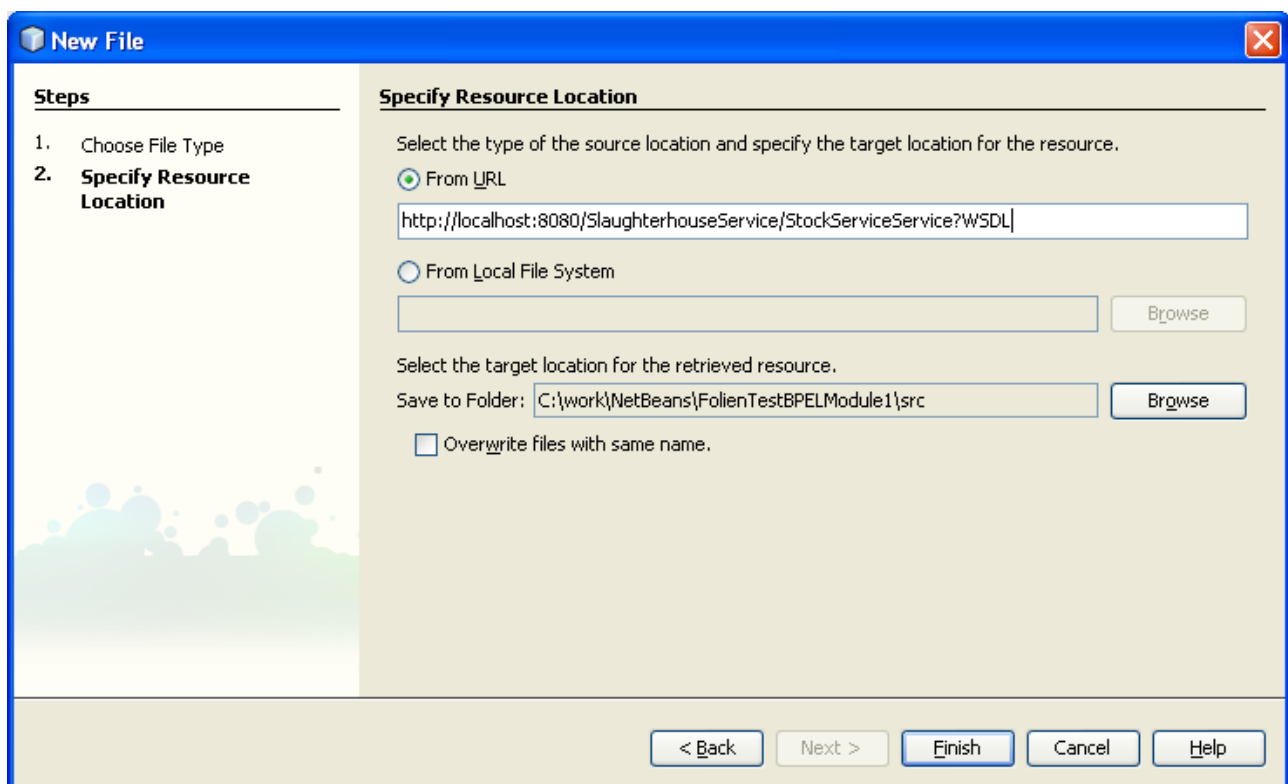


Abbildung 8: Externe WSDL-Datei hinzufügen

7. Nun fügen Sie dem AvailabilityProcess einen neuen Partner Link namens *StockService* hinzu, indem Sie aus der Palette die *Partnerlink* Aktivität auf die rechte Seite des Prozesses ziehen. Wählen Sie beim anschließenden Editieren die entsprechende WSDL-Datei aus (siehe Abbildung 9).

Create New Partner Link

Name:

WSDL File:

☐ Use Existing Partner Link Type

Partner Link Type:

My Role:

Partner Role:

☒ Use a Newly Created Partner Link Type

Create in File:

Partner Link Type Name:

☐ Process will implement (My Role)

Role Name:

Port Type:

☒ Partner service will implement (Partner Role)

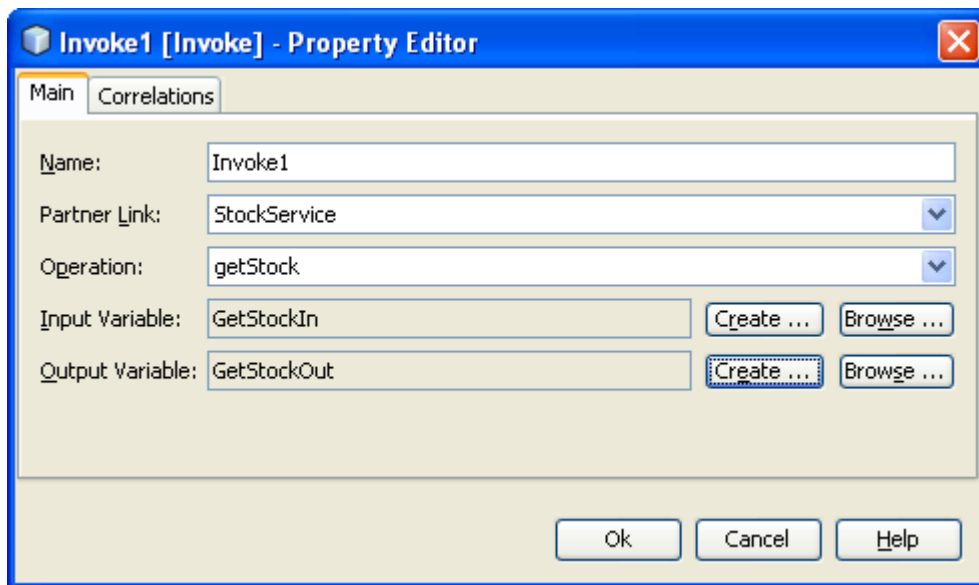
Role Name:

Port Type:

Abbildung 9: Erstellen des StockService PartnerLinks.

Übung: Aufrufen des StockServices

1. Fügen Sie dem Prozess nach *SetAvailabilityTrue* einen neuen Scope zu und nennen Sie ihn *CallStockService*.
2. Fügen Sie nun eine *Invoke* Aktivität mit dem Namen *StockServiceInvoke* hinzu.
3. Konfigurieren Sie das *StockServiceInvoke*, indem Sie den *StockService* als Partnerlink und dessen Funktion *getStock* als Operation setzen. Ferner müssen noch die beiden Variablen wie in Abbildung 11 gesetzt werden.

**Abbildung 10:** Konfigurieren des Invokes

4. Im nächsten Schritt fügen Sie nach der Invoke ein Assign namens *InitGetStockIn* dem Scope hinzu.
5. In dem zugehörigen Mapper weisen Sie *GetStockIn* einen String „Dummy“ zu.

Hinweis: Obwohl *getStock* keinen Wert erwartet, muss die Variable initialisiert werden, deswegen weisen wir ihr den Wert „Dummy“ zu.

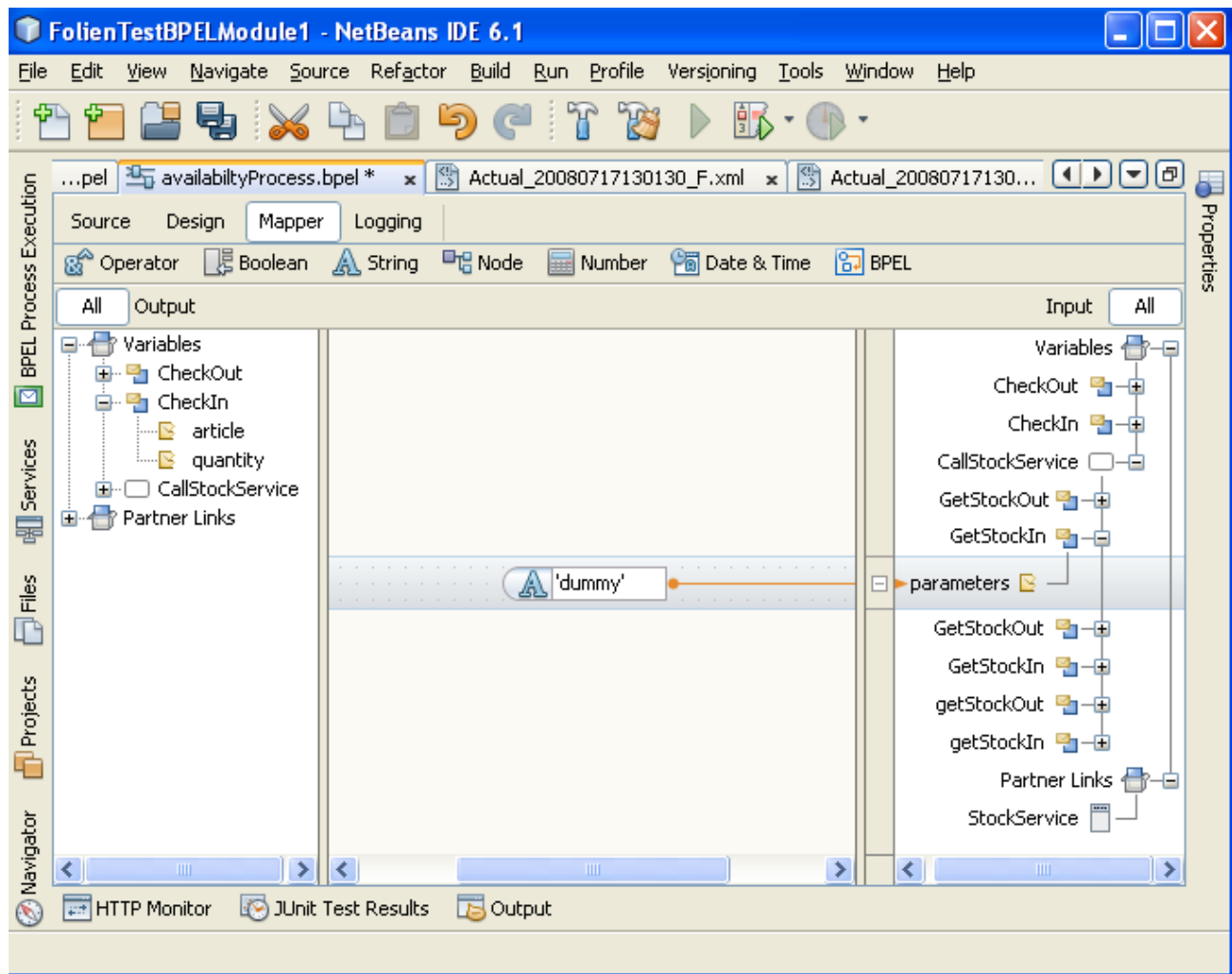


Abbildung 11: Konfigurieren des Assigns

6. Nun sollte der Prozess wie in folgender Abbildung aussehen:

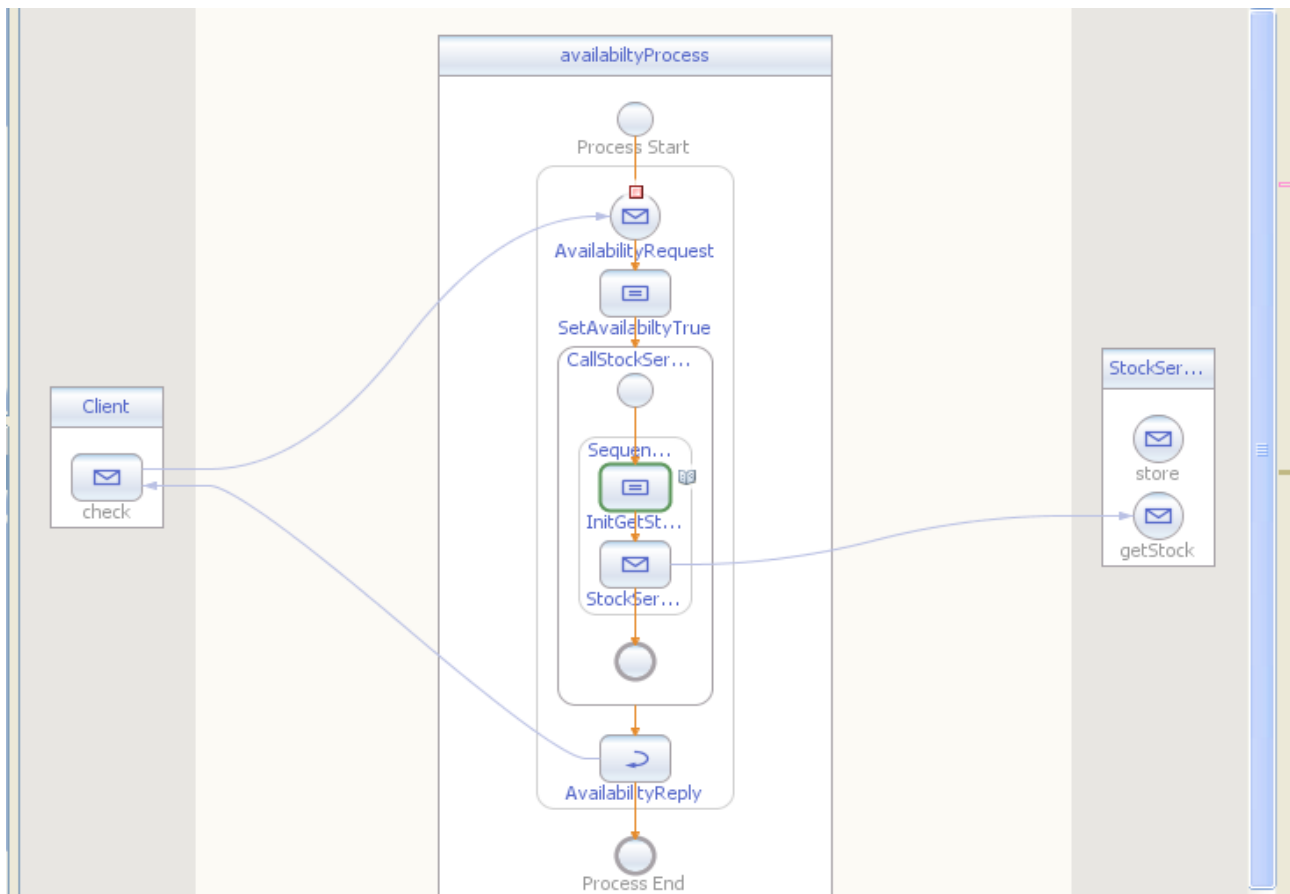


Abbildung 12: fertiger Prozess

7. Testen Sie den Prozess.

Übung: Availability berechnen

1. Fügen Sie dem Prozess eine globale Variable hinzu, indem Sie oberhalb des Prozesses auf „Add Variable...“ klicken (siehe Abbildung 13).

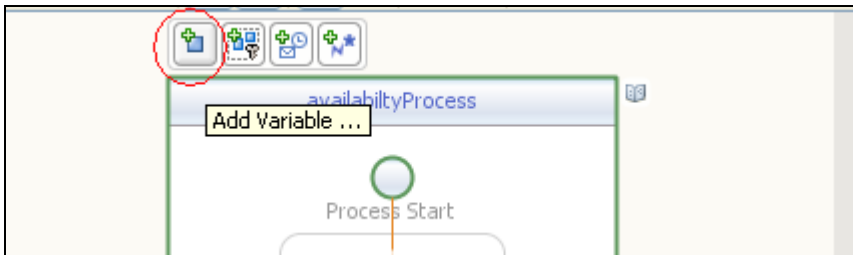


Abbildung 13: Variable hinzufügen

2. Nennen Sie die Variable *anzahl* und geben Sie ihr den Typ *int*.

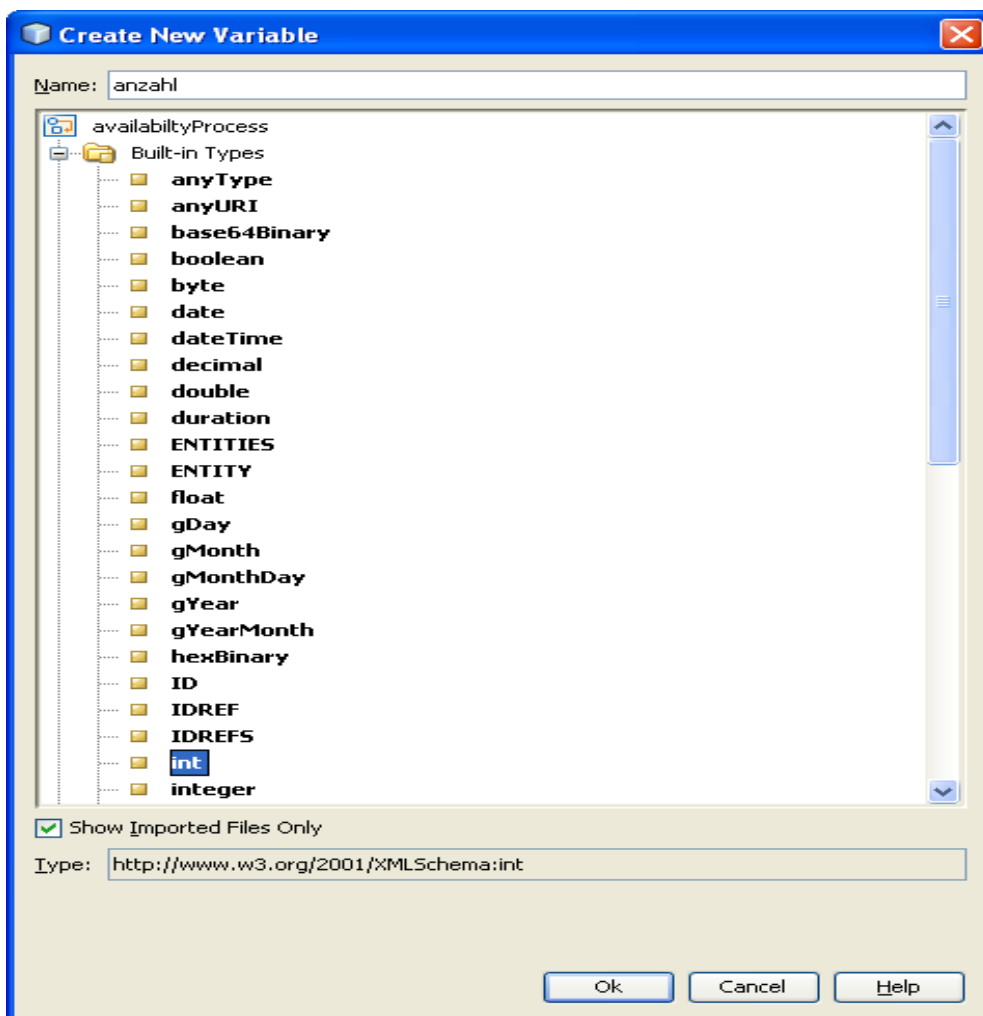


Abbildung 14: Variable erzeugen

3. Fügen Sie nun hinter *StockServiceInvoke* (noch im Scope) eine *Assign* Aktivität ein. Nennen Sie es *StockSum*.
4. Im zugehörigen Mapper wählen Sie auf der rechten Seite die *anzahl* Variable aus, fügen der grafischen Ansicht über *Number / Sum* einen Addierer hinzu und ziehen die Pfeile (wie in Abbildung 15) zwischen der *quantity* Variable, dem Addierer und der *anzahl* Variable.

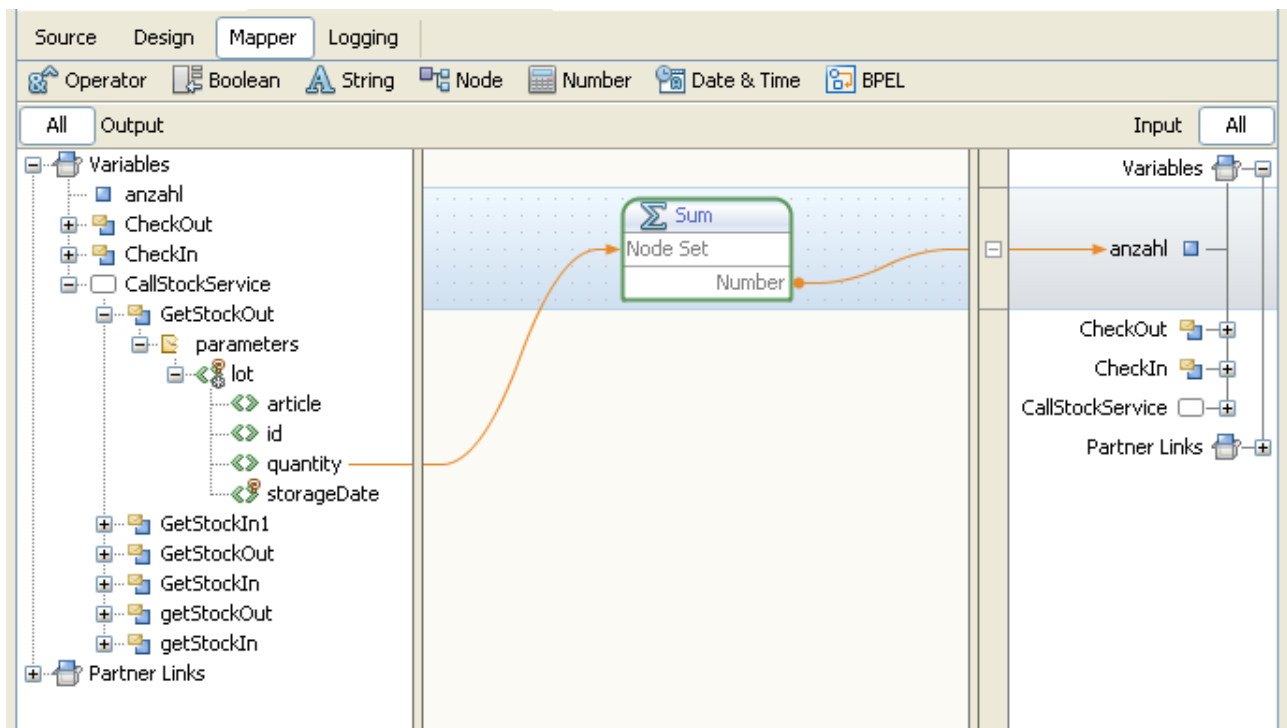


Abbildung 15: Addierer in der Assign Aktivität StockSum einbauen

5. Hinter die gerade erzeugte *Assign* Aktivität ziehen Sie nun eine *If* Aktivität.

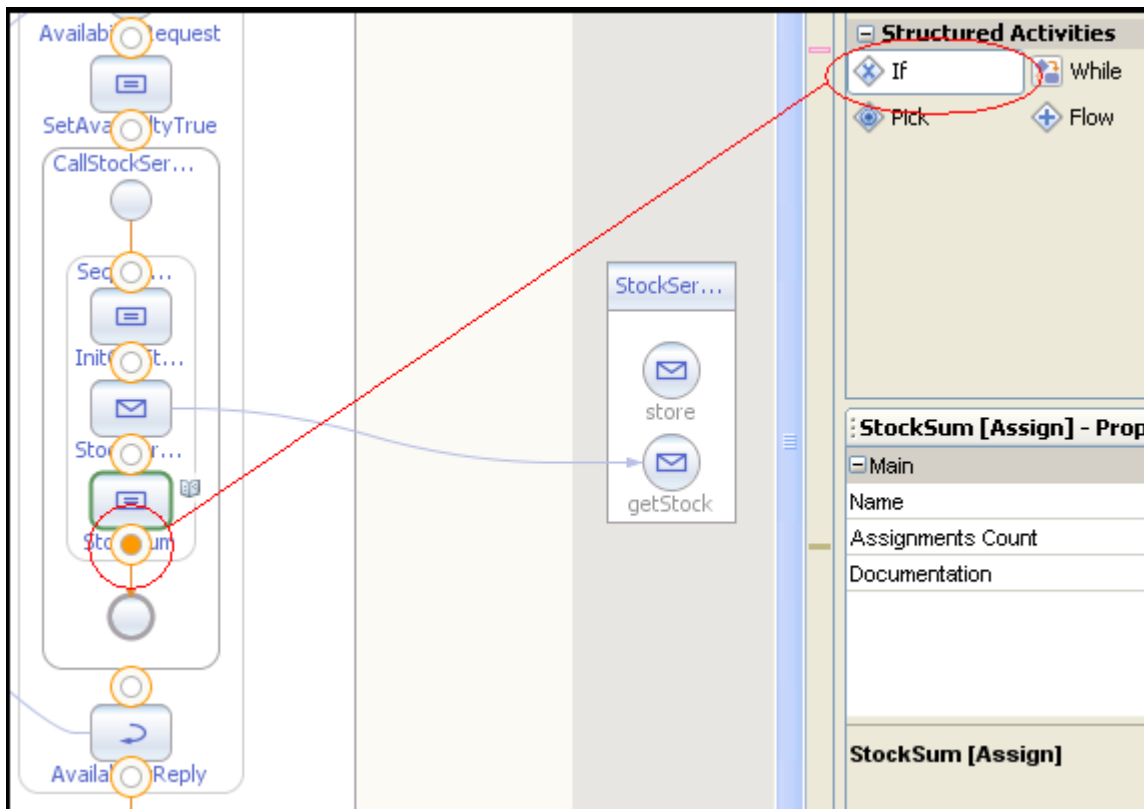


Abbildung 16: Prozess um *If* Aktivität erweitern

6. Mit einem Doppelklick auf die erste Verzweigung in der *If* Aktivität gelangen Sie in den zugehörigen Mapper. Klicken Sie hier zunächst rechts auf die „*Boolean Condition*“ und dann in dem XPath Funktion Menü auf *Operator / Greater or Equal*. Verbinden Sie nun die *anzahl* und die *quantity* Variablen, sowie die *Boolean Condition* mit dem *Greater or Equal* Operator (siehe Abbildung 17).

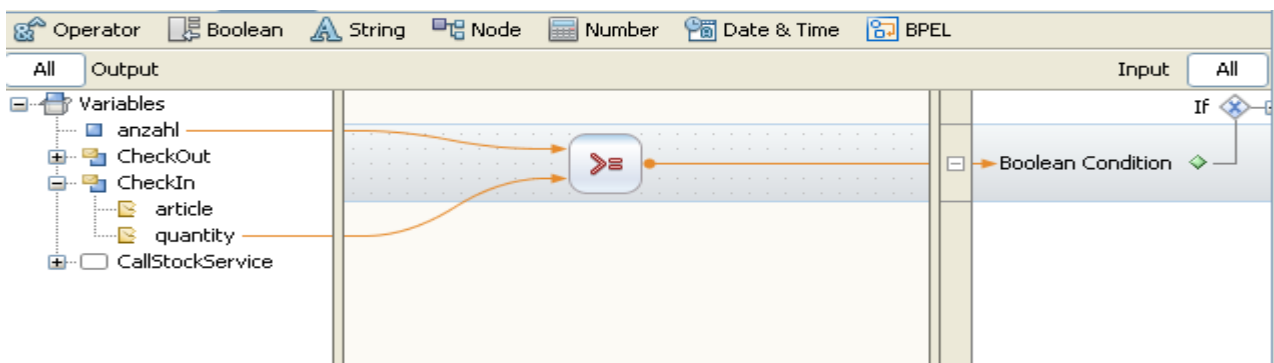


Abbildung 17: Mapper der *If* Bedingung

7. Nun fügen Sie zwei *Assign* Aktivitäten in die beiden Pfade der *If* Aktivität hinzu. Die linke *Assign* Aktivität können Sie zur besseren Anschauung in *If* umbenennen, die andere in *Else*.
8. Im Mapper von *If* klicken Sie auf auf der rechten Seite einmal auf die *availability* Variable unter *Variables / CheckOut*. Fügen Sie der grafischen Ansicht nun wieder ein *Logical True* über das Boolean Menü hinzu und verbinden Sie *availability* damit (Siehe Abbildung 18).

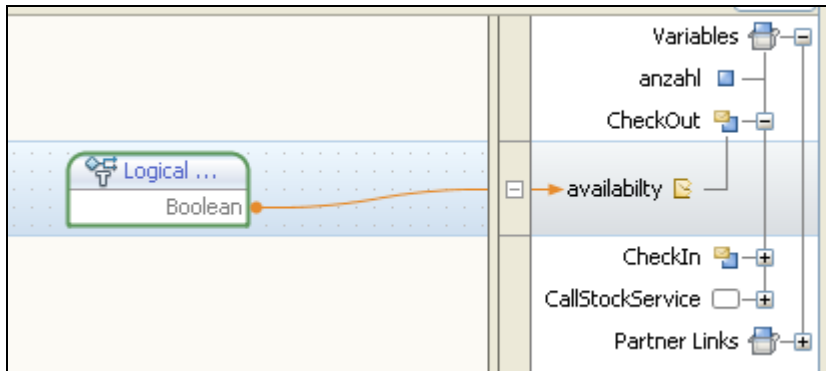


Abbildung 18: Mapper des *If* Assigns

9. Im Mapper von *Else* wiederholen wir das ganze, verwenden aber anstatt dem *Logical True* ein *Logical False* (siehe Abbildung 19).

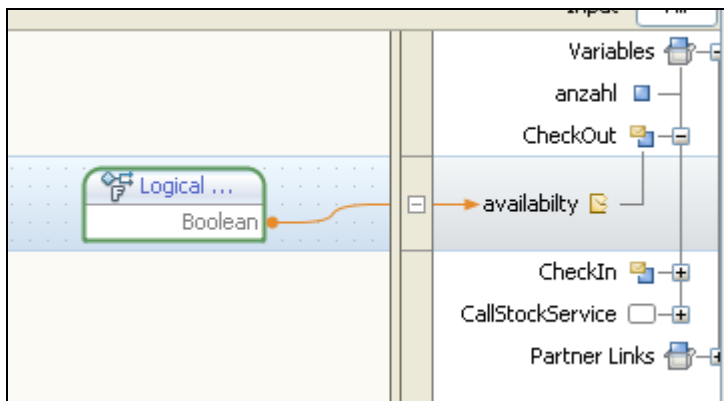


Abbildung 19: Mapper des *Else* Assigns

10. Für die Funktionalität des Prozesses braucht man nun nicht mehr die alte *SetAvailabilityTrue* Assign Aktivität. Diese können Sie unbesorgt löschen.

11. Der Prozess ist nun fertig erstellt. Abbildung 20 zeigt den Prozess in der Gesamtansicht.

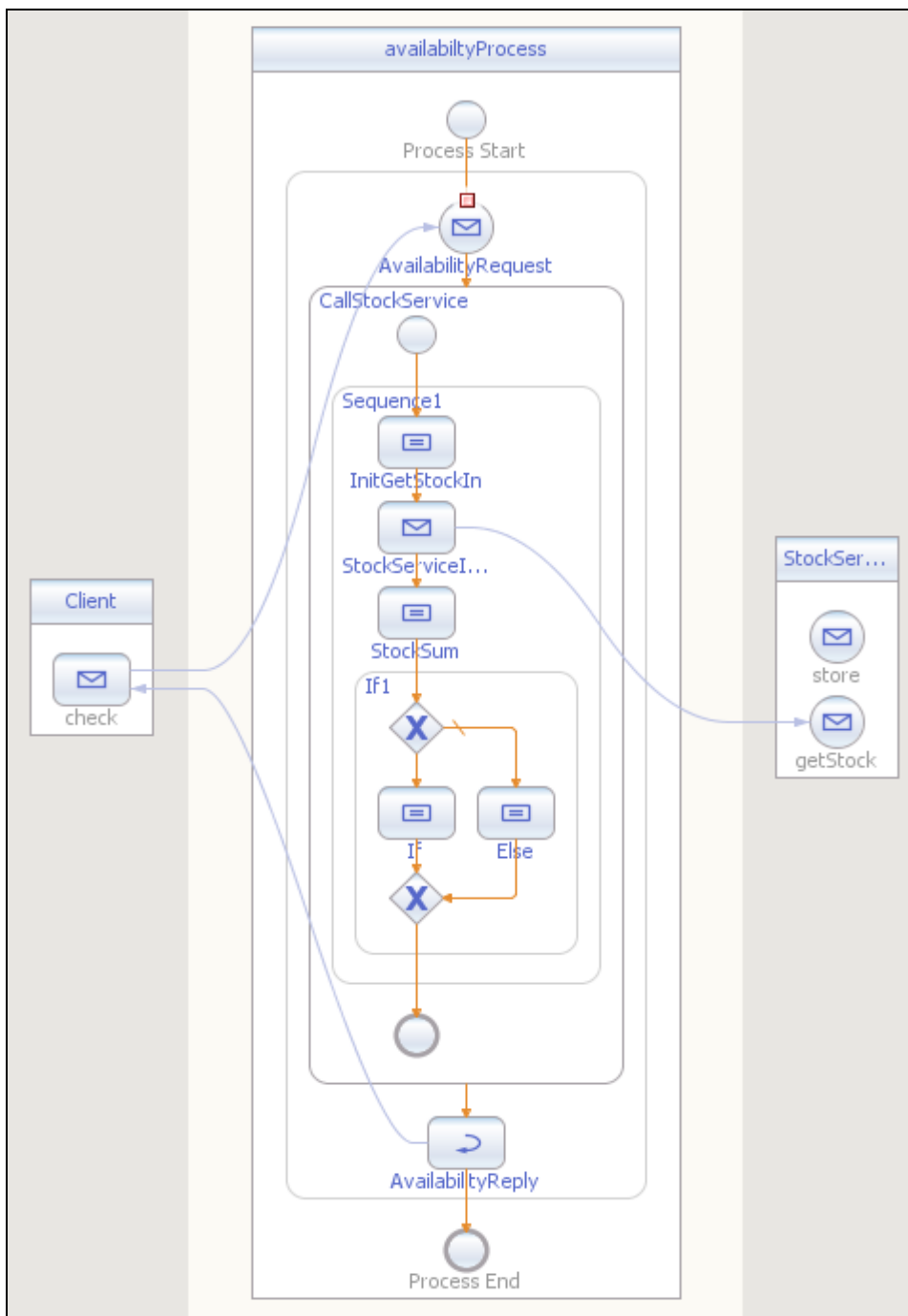


Abbildung 20: Gesamtansicht des Prozesses

Zusatzübung:

Wie könnte man sich die *If* Aktivität sparen und die Verfügbarkeit in einem Assign errechnen?

12. Testen Sie den Prozess.

Frage:

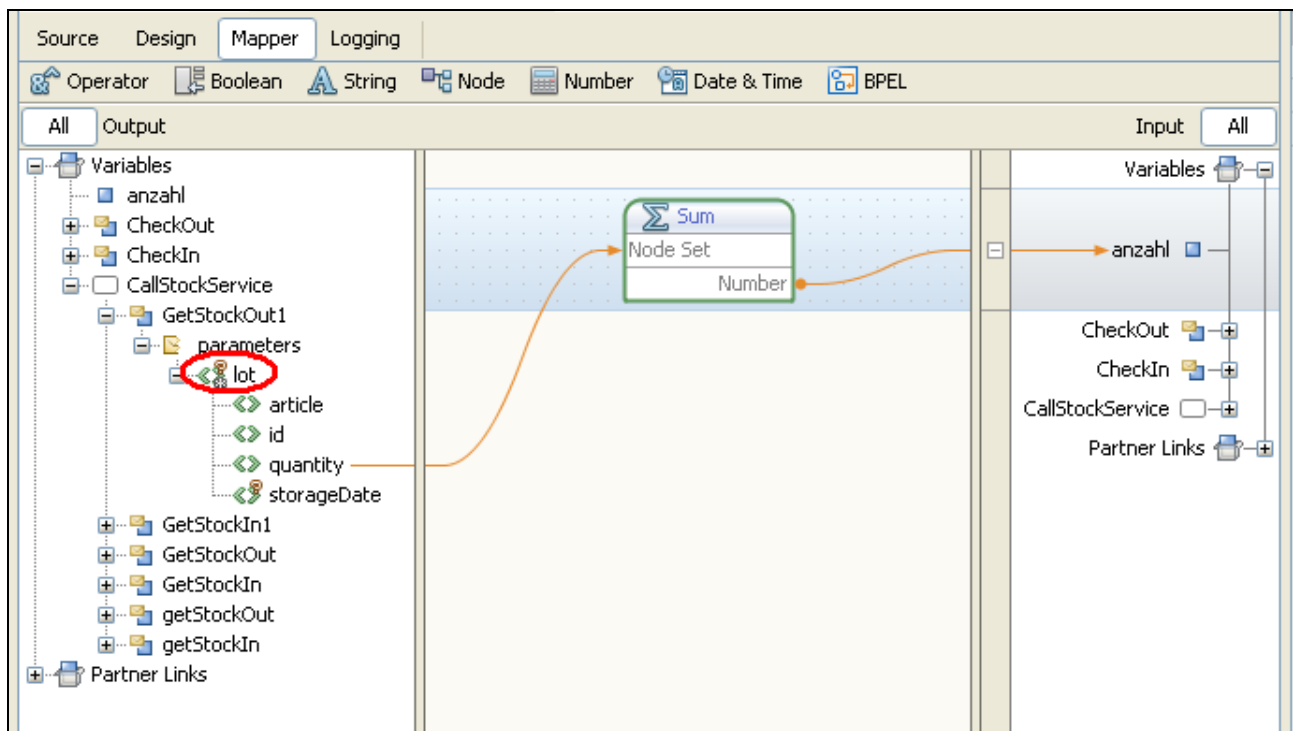
Warum gibt der Prozess *availability* als True zurück? Eigentlich hätte er False zurückgeben müssen, da wir nach 100 Stück von Artikel 1 gefragt haben, das *Slaughterhouse* aber nur 50 Stück gelagert hat!

Übung: Fehler suchen

1. Debuggen Sie den Prozess und beobachten Sie unten rechts in der *Local Variables View* die *anzahl* Variable.

Übung: Fehler korrigieren

1. Anscheinend liegt der Fehler in der Zuweisung des falschen Wertes an die Variable *anzahl* im *StockSum* Assign. Hier werden fälschlicherweise die Anzahl aller Artikel im Lager der Variable zugewiesen.
2. Um dies zu korrigieren gehen Sie auf den Mapper von *StockSum*.
3. Nun klicken Sie rechts auf die *lot* Variable (siehe Abbildung 21) und gelangen durch „Add Predicate...“ in den Prädikat Editor.

**Abbildung 21:** Prädikat hinzufügen

4. Im Prädikat Editor klicken Sie rechts auf das Prädikat und fügen nun über *Operator / Equal* einen Vergleichsoperator hinzu. Verglichen werden soll die *CheckIn article Variable* mit dem Kindelement *article* der *GetStockOut* Variable (siehe Abbildung 22).

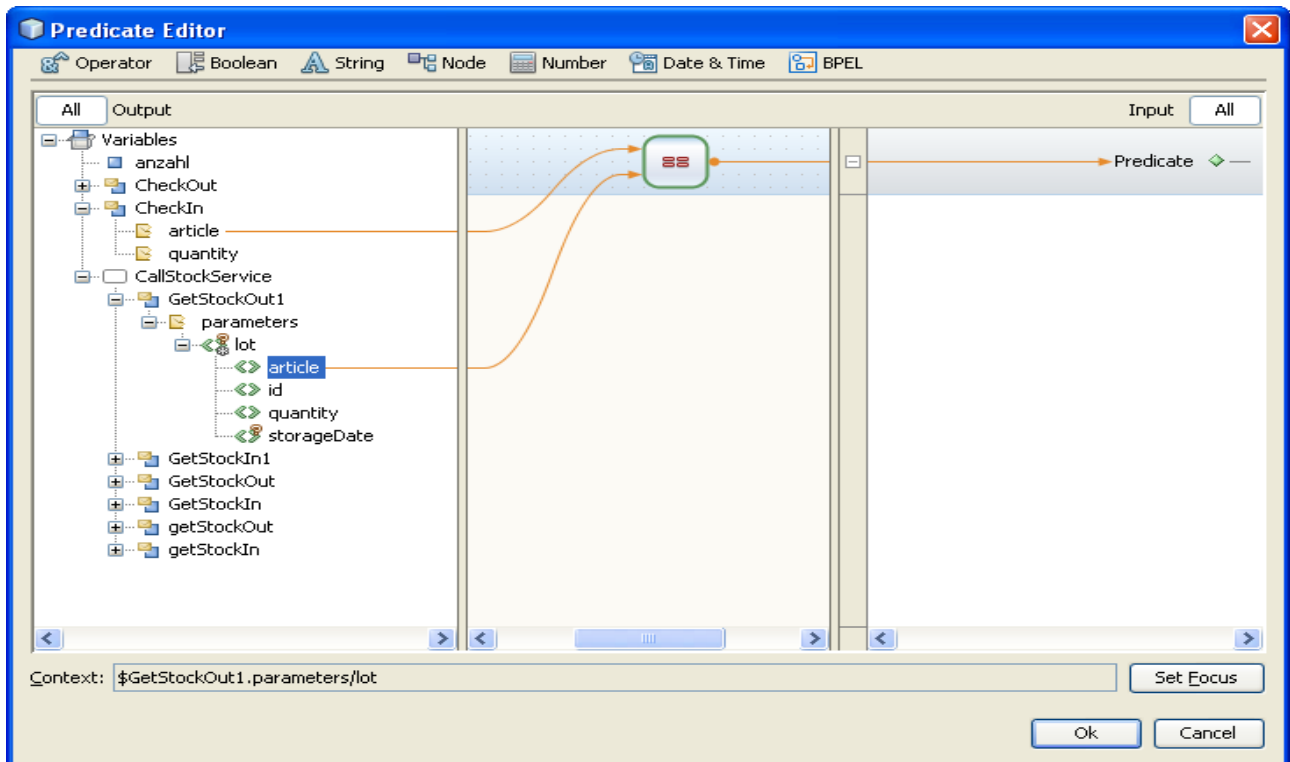


Abbildung 22: Prädikat Editor

- Nachdem Sie mit *Okay* das Prädikat erstellt haben. Müssen Sie im Mapper die *quantity* Variable des bisherigen Lots mit dem neu modifizierten Lot ersetzen(siehe Abbildung 23).

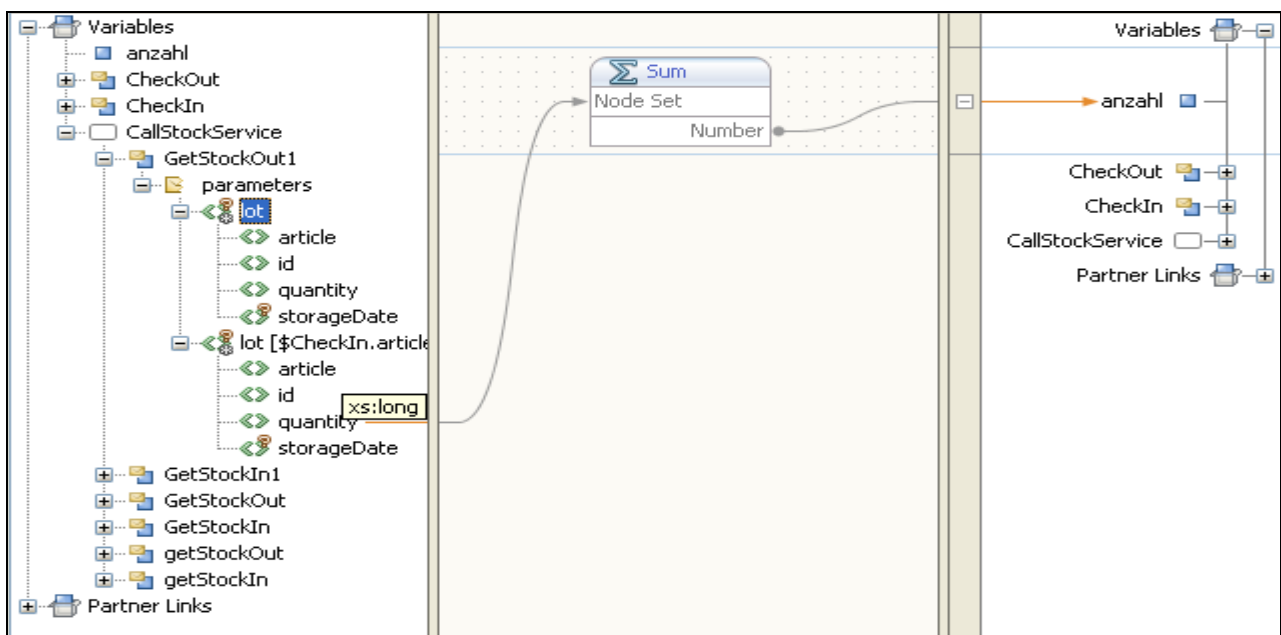
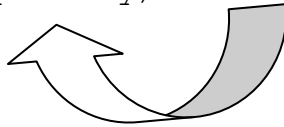


Abbildung 23: StockSum korrigieren

6. Testen Sie erneut den Prozess.

Hinweis: Sollten Sie eine Fehlermeldung bekommen, nachdem Sie das Prädikat hinzugefügt haben versuchen Sie die beiden Operatoren von der = Operation zu vertauschen.

```
sum($GetStockOut1.parameters/lot[article  
=$CheckIn.article]/quantity)
```



5. Der OrderProcess

5.1. Vorbereitung

Öffnen Sie die Kursdisk und kopieren Sie von *SlaughterhouseDateien* das *order.xsd* Schema in das Projekt.

Übung: OrderService WSDL erstellen

1. Erstellen Sie im Projekt *SlaughterhouseProcess* ein WSDL Dokument *OrderService.wsdl*.
2. Ändern Sie den Target Namespace in:

`http://predic8.com/OrderService`

3. Ändern Sie den WSDL Type in *concrete WSDL Document*
4. Wählen Sie *SOAP Binding* und *RPC Literal Type*.

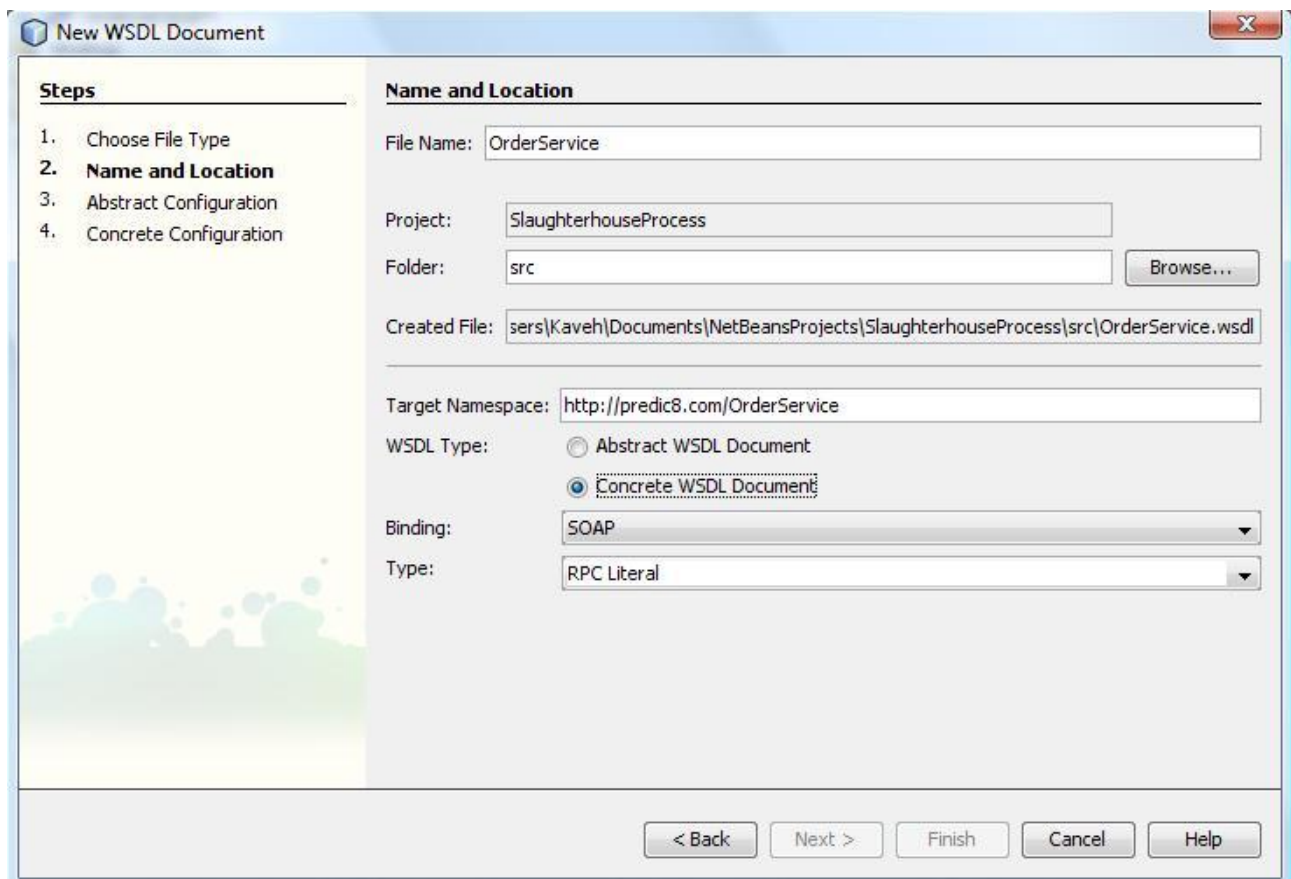


Abbildung 24: Neues WSDL Dokument erstellen

5. Klicken Sie auf den Button *Next*.
6. Ändern Sie den Namen der Operation in *placeOrder*
7. Ändern Sie den Namen der Input Message auf *parameters* und wählen Sie den Complex Type *placeOrderType* des *order* Schemas.

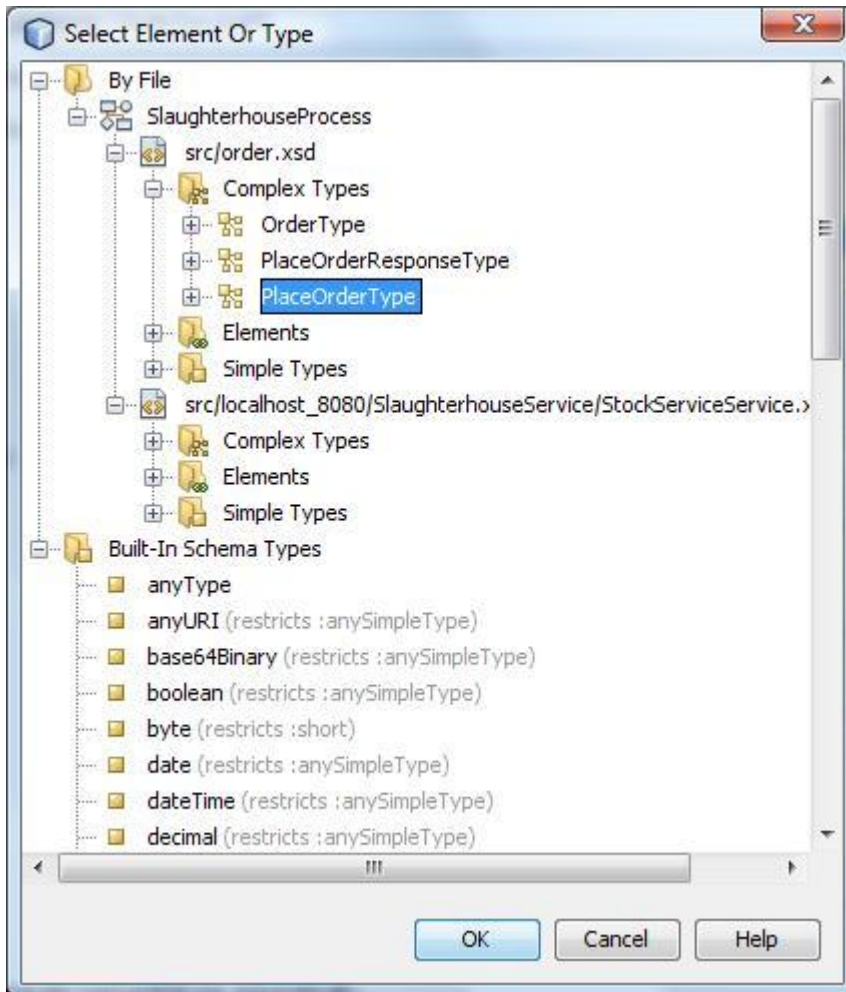


Abbildung 25: Datentyp der Input Message wählen

8. Ändern Sie den Namen der *Output Message* auf *parameters* und wählen Sie den Type *placeOrderResponseType* des *order* Schemas.

New WSDL Document

Steps

1. Choose File Type
2. Name and Location
- 3. Abstract Configuration**
4. Concrete Configuration

Abstract Configuration

Port Type Name:

Operation Name:

Operation Type:

Input:

Message Part Name	Element Or Type
parameters	ns:PlaceOrderType

Output:

Message Part Name	Element Or Type
parameters	ns:PlaceOrderResponseType

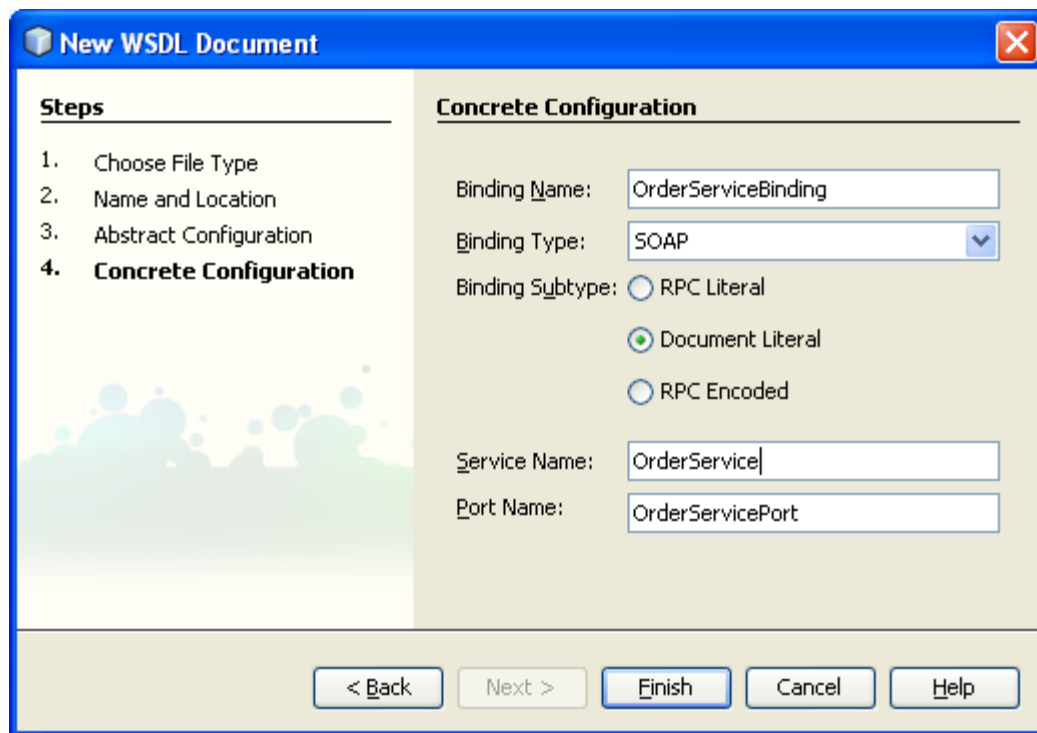
Fault:

Message Part Name	Element Or Type
-------------------	-----------------

☒ Generate partnerlinktype automatically.

Abbildung 26: Konfigurieren des WSDL Dokuments

9. Klicken Sie auf den Button *Next* und dann auf *Finish*.



The screenshot shows a dialog box titled "New WSDL Document" with a close button (X) in the top right corner. The dialog is divided into two main sections: "Steps" on the left and "Concrete Configuration" on the right. The "Steps" section lists four steps: 1. Choose File Type, 2. Name and Location, 3. Abstract Configuration, and 4. **Concrete Configuration**. The "Concrete Configuration" section contains several input fields and radio buttons. The "Binding Name" field is set to "OrderServiceBinding". The "Binding Type" dropdown menu is set to "SOAP". The "Binding Subtype" section has three radio buttons: "RPC Literal" (unselected), "Document Literal" (selected), and "RPC Encoded" (unselected). The "Service Name" field is set to "OrderService" and the "Port Name" field is set to "OrderServicePort". At the bottom of the dialog, there are five buttons: "< Back", "Next >", "Finish", "Cancel", and "Help".

Abbildung 27: Konfiguratiuon des WSDL Dokuments

Übung: OrderProcess erstellen

1. Erstellen Sie im Projekt *SlaughterhouseProcess* einen neuen BPEL Prozess *OrderProcess*.
2. Ändern Sie den Target Namespace in:

`http://predic8.com/OrderProcess`

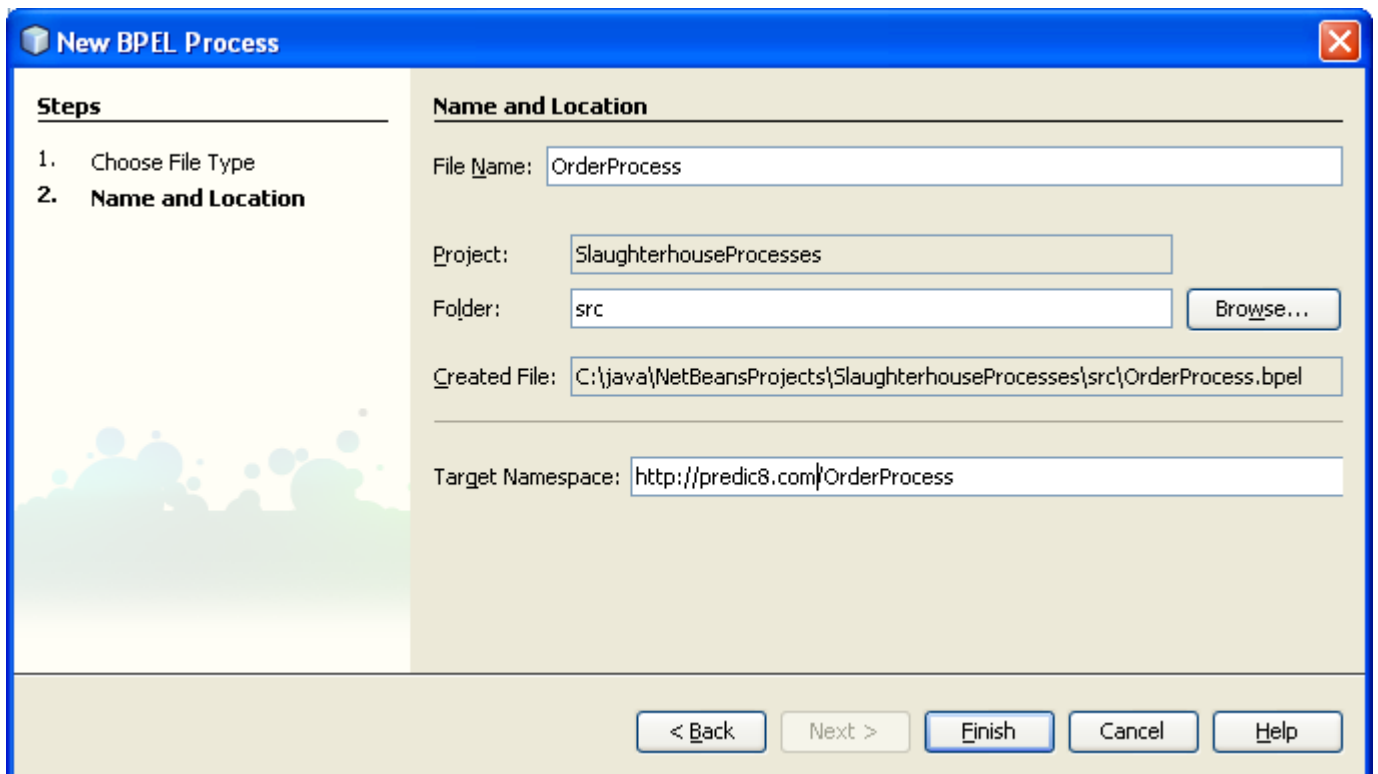


Abbildung 28: Erstellen eines neuen BPEL Prozesses

- Fügen Sie dem Prozess einen neuen Partner Link *OrderProcessClient* hinzu, welcher auf den *OrderService* verweist.

Create New Partner Link

Name:

WSDL File:

☒ Use Existing Partner Link Type

Partner Link Type:

My Role:

Partner Role:

☐ Use a Newly Created Partner Link Type

Create in File:

Partner Link Type Name:

☐ Process will implement (My Role)

Role Name:

Port Type:

☐ Partner service will implement (Partner Role)

Role Name:

Port Type:

Abbildung 29: Neuen Partner Link erstellen

- Bestätigen Sie mit OK.

5. Fügen Sie dem Prozess eine neue Receive Aktivität hinzu. Klicken Sie hierzu in der Komponentenpalette auf Receive und ziehen Sie diese auf den Konnektor in der Design-Ansicht.

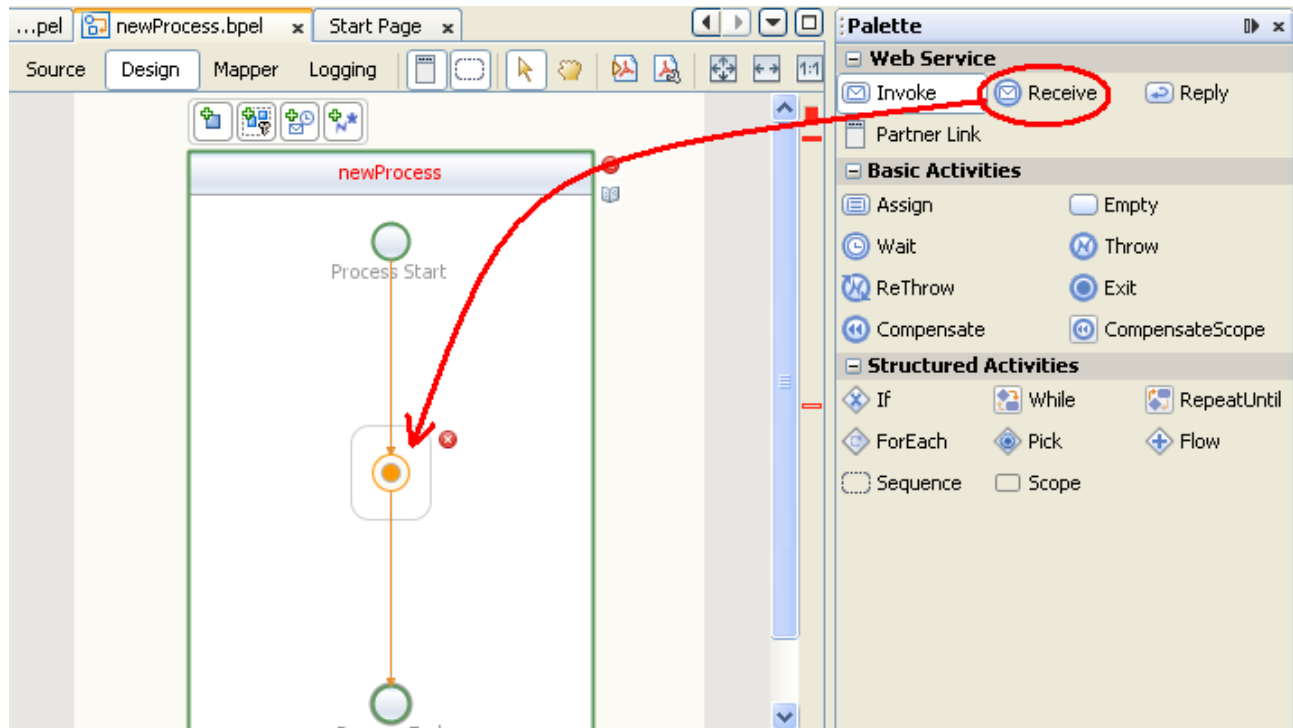


Abbildung 30: Hinzufügen einer Komponente

6. Konfigurieren Sie die neue Aktivität wie in folgendem Screenshot gezeigt:

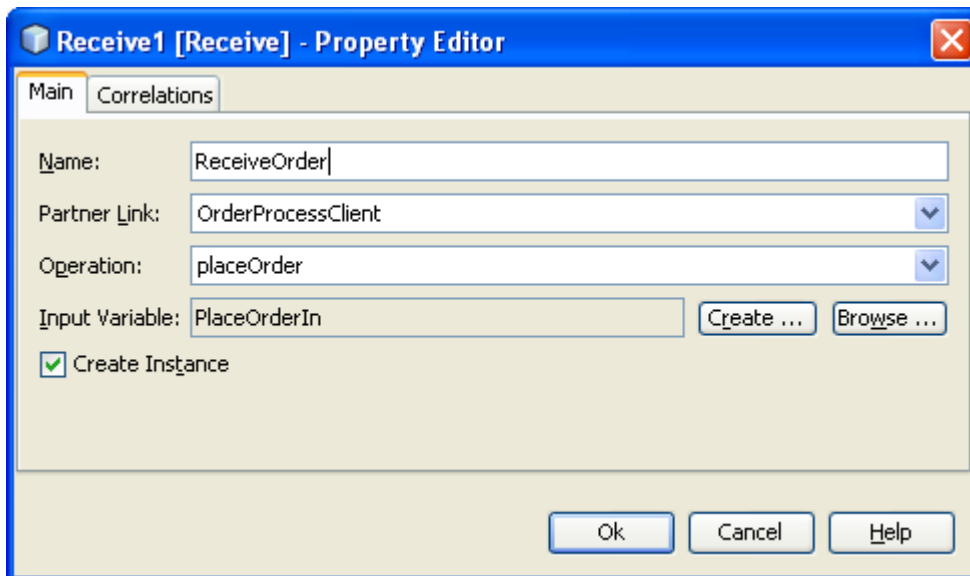


Abbildung 31: Konfigurieren der Receive Aktivität

Hinweis: Die Input Variable muss über den Button *Create...* auf Prozessebene neu erstellt werden.

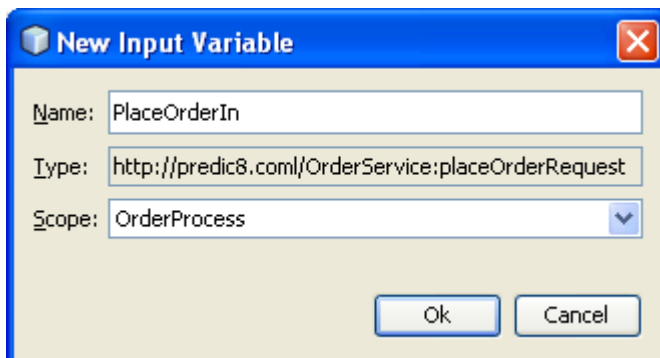


Abbildung 32: Erstellen einer neuen Variable

7. Fügen Sie dem Prozess nach dem Aufruf der *Receive* Aktivität eine *Reply* Aktivität hinzu.

8. Konfigurieren Sie die neue Aktivität wie in folgendem Screenshot gezeigt, auch hier muss die benötigte Variable auf Prozessebene neu erstellt werden.

Reply1 [Reply] - Property Editor

Main Correlations

Name: ReplyToPlaceOrder

Partner Link: OrderProcessClient

Operation: placeOrder

☒ Normal Response

Output Variable: PlaceOrderOut Create ... Browse ...

☐ Fault Response

Fault Name: Choose ...

Fault Variable: Create ... Browse ...

Ok Cancel Help

Abbildung 33: Konfigurieren der Reply Komponente

Ihr Prozess sieht jetzt wie in dem folgenden Screenshot gezeigt aus:

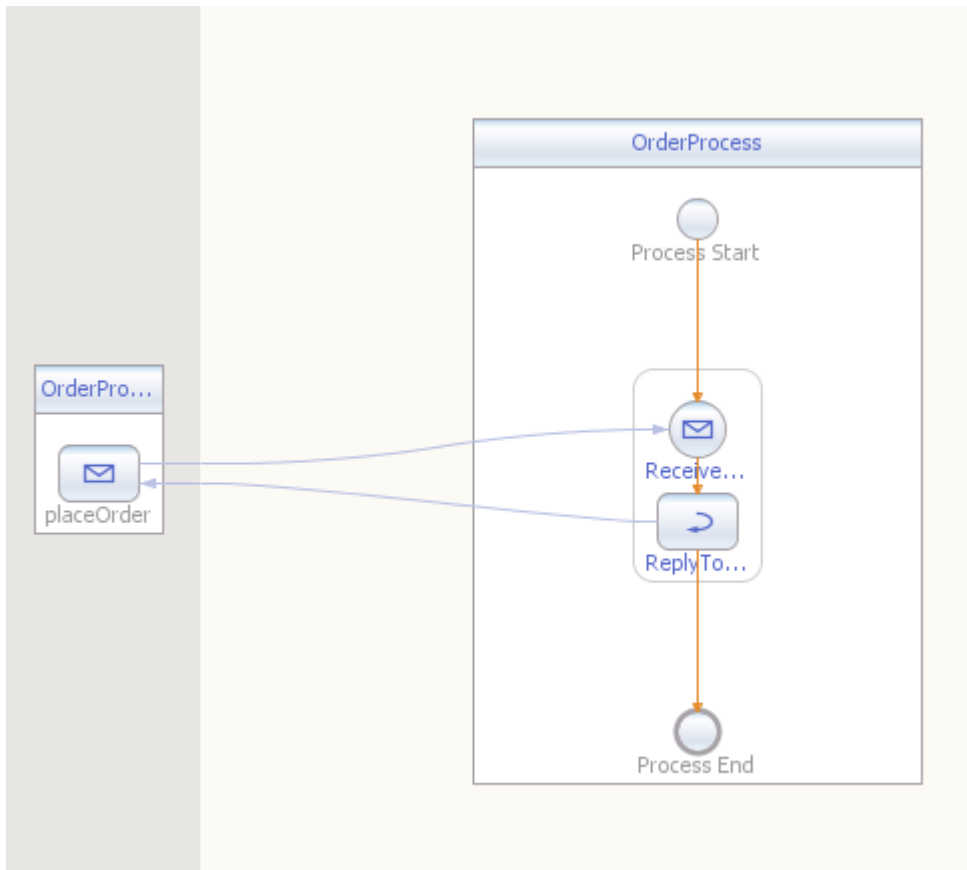


Abbildung 34: Der Prozess OrderProcess

6. Einbinden eines bestehenden Web Service

6.1. Vorbereitung

1. Betrachten Sie den Web Service *ProductionService* im Projekt *Slaughterhouse*.
2. Führen Sie ein Deployment des Services durch und lassen Sie sich das WSDL Dokument anzeigen, indem Sie ein Rechtsklick auf den Service machen und dann *Test Webservice* wählen.

Übung: Einbinden des Web Service

1. Fügen Sie dem Projekt *SlaughterhouseProcess* ein neues externes WSDL Dokument hinzu.
2. Verweisen Sie auf das WSDL Dokument des *ProductionService*.

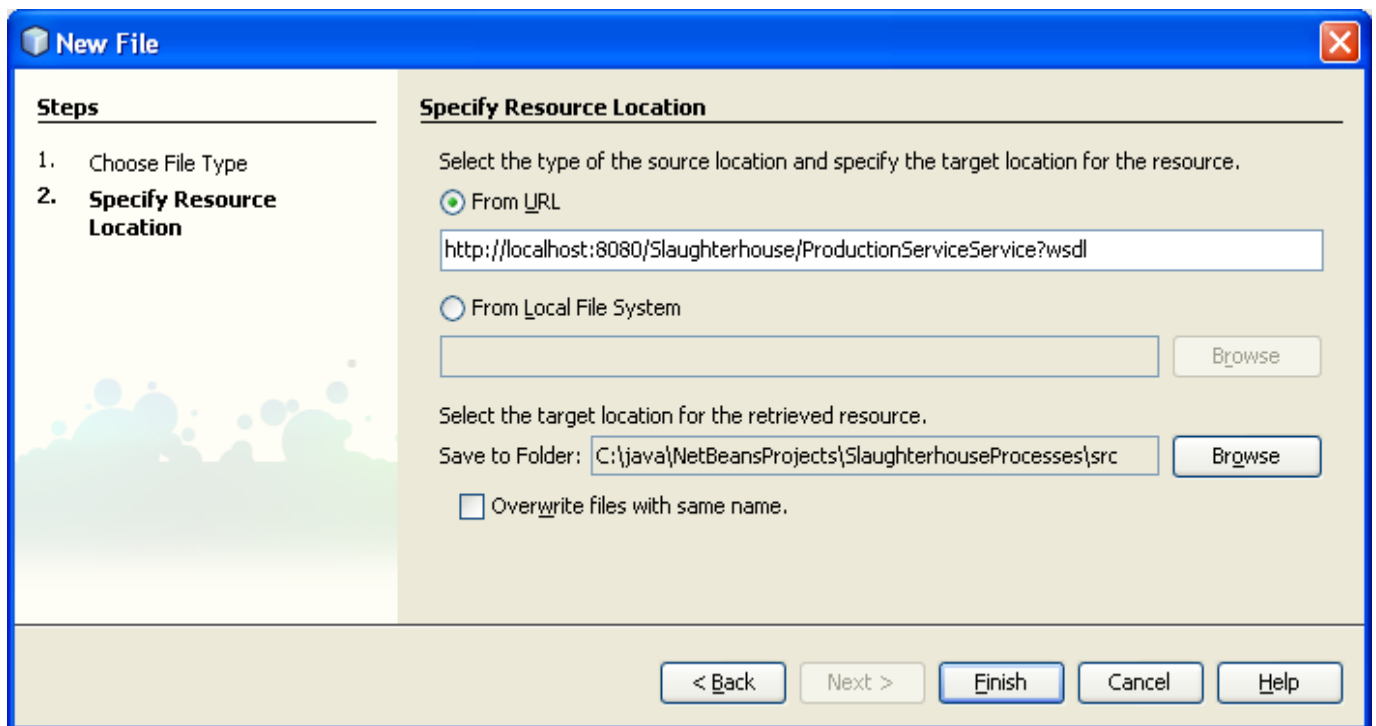


Abbildung 35: Neues externes WSDL Dokument einbinden

3. Fügen Sie zwischen der Recieve und der Reply Aktivität ein *Sequence* Element ein.
4. Fügen Sie dem Prozess einen neuen Partner Link *ProductionServicePL*, der auf den *ProductionService* verweist hinzu.
5. Nennen Sie den Partner Link Type *ProductionServicePLT*.

Create New Partner Link

Name:

WSDL File:

☐ Use Existing Partner Link Type

Partner Link Type:

My Role:

Partner Role:

☒ Use a Newly Created Partner Link Type

Create in File:

Partner Link Type Name:

☐ Process will implement (My Role)

Role Name:

Port Type:

☒ Partner service will implement (Partner Role)

Role Name:

Port Type:

Abbildung 36: Erstellen eines neuen Partner Link

6. Setzen Sie ein Scope Element um die Sequenz, indem Sie einen Rechtsklick auf das Sequence Element ausführen und *Wrap with/Scope* wählen.
7. Fügen Sie der Sequenz eine Invoke Aktivität hinzu und konfigurieren Sie diese wie im Screenshot gezeigt. Erstellen Sie die Input und die Output Variable auf Ebene des vorher hinzugefügten Scopes.

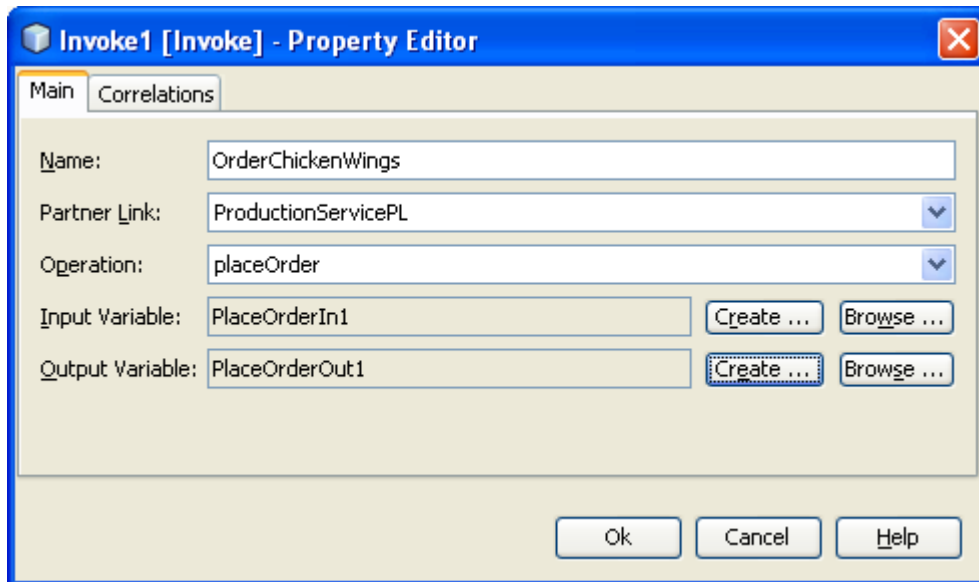


Abbildung 37: Konfiguration der Invoke Aktivität

8. Fügen Sie vor der Invoke Aktivität eine Assign Aktivität ein.
9. Expandieren Sie in der Mapper-Darstellung die Parameter der in Scope1 enthaltenen Variablen *PlaceOrderIn1*.
10. Weisen Sie dem Parameter *article* den Wert 0 und dem Parameter *quantity* den Wert 70 zu.

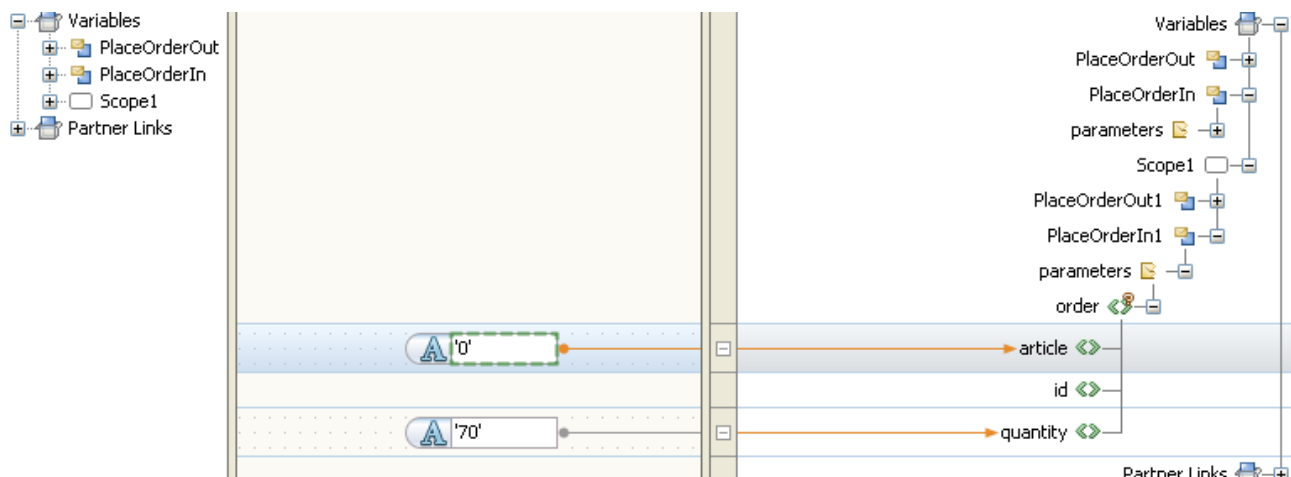


Abbildung 38: Mapper-Ansicht

11. Fügen Sie eine weitere Assign Aktivität hinter der Invoke Aktivität aber außerhalb des Scope Elementes hinzu.
12. Setzen Sie die *orderId* der globalen Variable *placeOrderOut* auf den Wert 537.

Übung: Testen des Prozesses

1. Fügen Sie, sofern noch nicht geschehen, dem Projekt *SlaughterhouseApp* den erstellten Prozess als JBI Modul hinzu.
2. Führen Sie ein Deployment des Projektes *SlaughterhouseApp* durch.
3. Fügen Sie dem Projekt einen neuen Test hinzu, wählen Sie das WSDL Dokument des *OrderService* und führen Sie den Test auf die Methode *placeOrder* aus.
4. Betrachten Sie die Antwort *output.xml* des Service und die Ausgabe in der Glassfish Konsole.

Übung: Bestellen von mehreren Artikeln

1. Setzen Sie um das Scope Element ein Sequence Element.
2. Wählen Sie das Scope Element aus und kopieren Sie dieses mit Strg+C.
3. Nach dem Kopieren können Sie den Scope durch Anklicken des Konnektors hinter dem Scope Element einfügen.

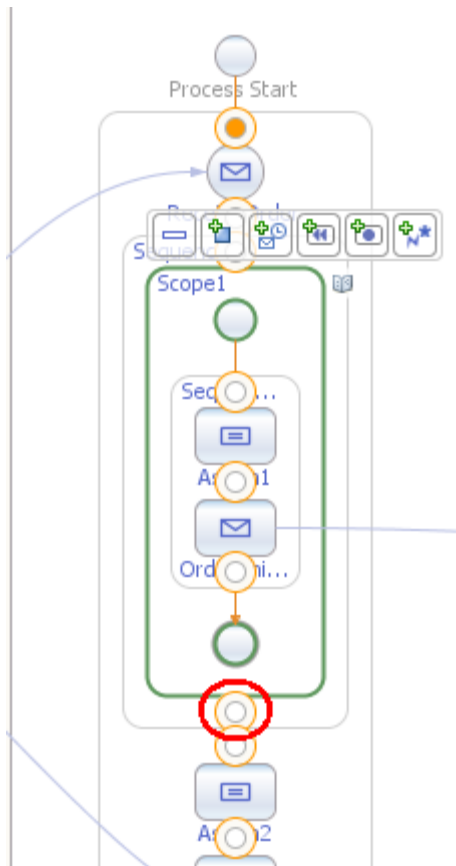


Abbildung 39: Einfügen eines kopierten Elementes

4. Wiederholen Sie den Vorgang, so dass Sie innerhalb der Sequenz drei Bestellungen abgeben.
5. Benennen Sie die beiden hinzugefügten Scopes in Scope2 und Scope3 um.
6. Ändern Sie für Scope2 die Werte der Variable *article* auf 1 und *quantity* auf 100.
7. Ändern Sie für Scope3 die Werte der Variable *article* auf 2 und *quantity* auf 30.
8. Testen Sie den Prozess und betrachten Sie die Ausgabe in der Glassfish Konsole.

Frage: Wie lange dauerte die Ausführung?

Übung: Vertauschen der Reihenfolge innerhalb einer Sequenz

1. Klicken Sie rechts auf die Sequenz und wählen Sie *Change Order...*
2. In dem sich öffnenden Dialog kann mit Hilfe der Buttons *Move up* und *Move down* die Reihenfolge geändert werden.

Übung: Parallelisieren

1. Setzen Sie um das Element *Scope1* eine *Flow* Aktivität.
2. Wählen Sie das Element *Scope2* aus und schneiden Sie dieses mit Strg+x aus.
3. Fügen Sie das Element durch anklicken des entsprechenden Konnektors innerhalb der *Flow* Aktivität neben dem Element *Scope1* ein.
4. Wiederholen Sie den Vorgang für das Element *Scope3*.

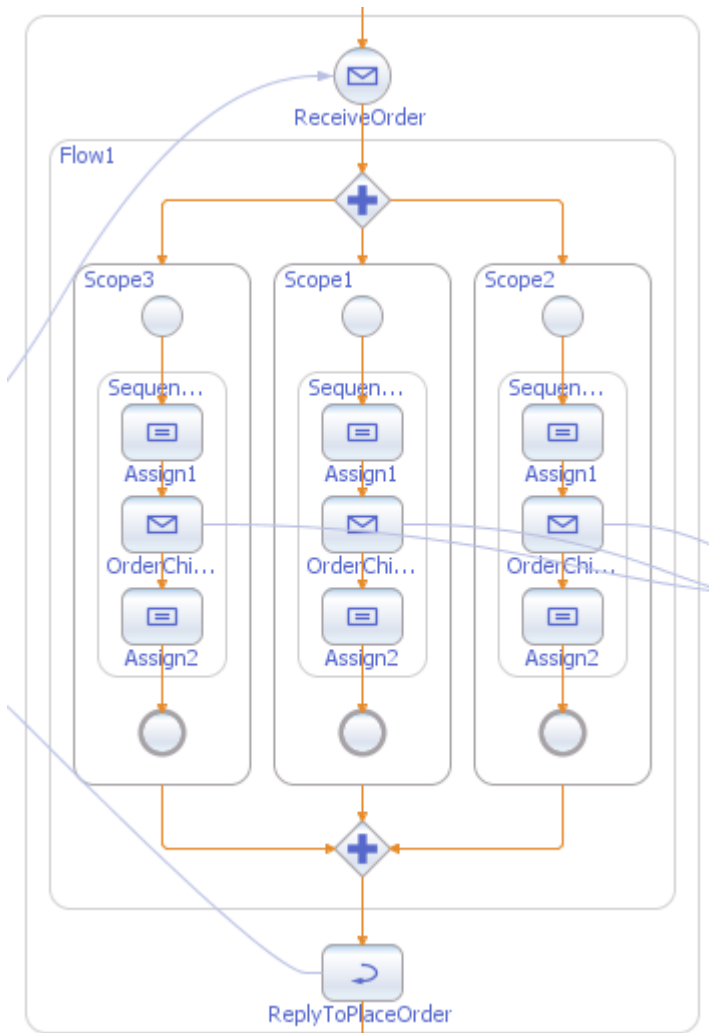


Abbildung 40: Die Flow Aktivität

5. Testen Sie den Prozess.
6. Nun setzen Sie ein Breakpoint auf *RecieveOrder* und führen Sie den Test diesmal in Debug-Modus aus.

Übung: Bestellung aufsplitten

1. Schneiden Sie das Element *Scope1* aus und fügen Sie dieses oberhalb der Flow Aktivität und innerhalb des Elementes *Sequence2* ein.
2. Entfernen Sie die komplette Flow Aktivität.
3. Setzen Sie um das Element *Scope1* eine *ForEach* Aktivität.
4. Benennen Sie die ForEach Aktivität in *LoopLineItems* um.
5. Führen Sie einen Doppelklick auf die ForEach Aktivität aus, um in die Mapper-Ansicht zu gelangen.
6. Weisen Sie die Werte wie in folgendem Screenshot gezeigt zu:

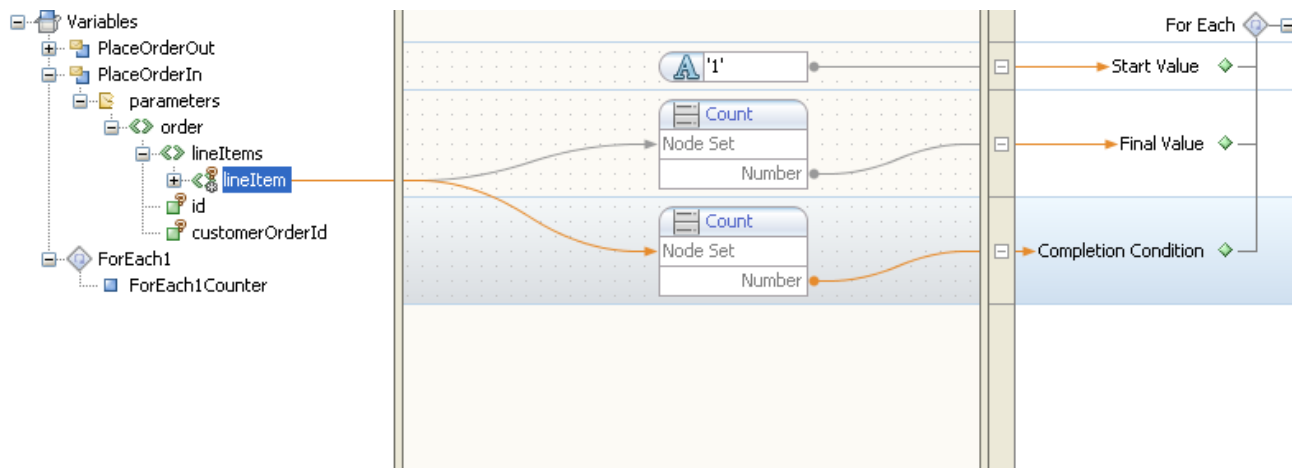


Abbildung 41: Zuweisen der Werte der ForEach Aktivität

7. Öffnen Sie den Dialog zum bearbeiten der Eigenschaften der ForEach Aktivität.
8. Ändern Sie den Wert des Attributes *Counter Variable Name* in *position*.

Main	
Name	LoopLineItems
Counter Variable Name	position
Start Counter Value	'1'
Final Counter Value	count(\$PlaceOrderIn.part1/ns0:order/ns...
Completion Condition	count(\$PlaceOrderIn.part1/ns0:order/ns...
Count Completed Branches Only	<input type="checkbox"/>
Documentation	

Counter Variable Name ?

Close

Abbildung 42: Bearbeiten der ForEach Aktivität

9. Öffnen Sie die Mapper-Ansicht der Assgin1 Aktivität.
10. Führen Sie einen Rechtsklick auf *lineltem* der globalen Variablen *PlaceOrderIn* aus und wählen Sie *AddPredicate...*
11. Konfigurieren Sie das Prädikat wie in folgendem Screenshot gezeigt.

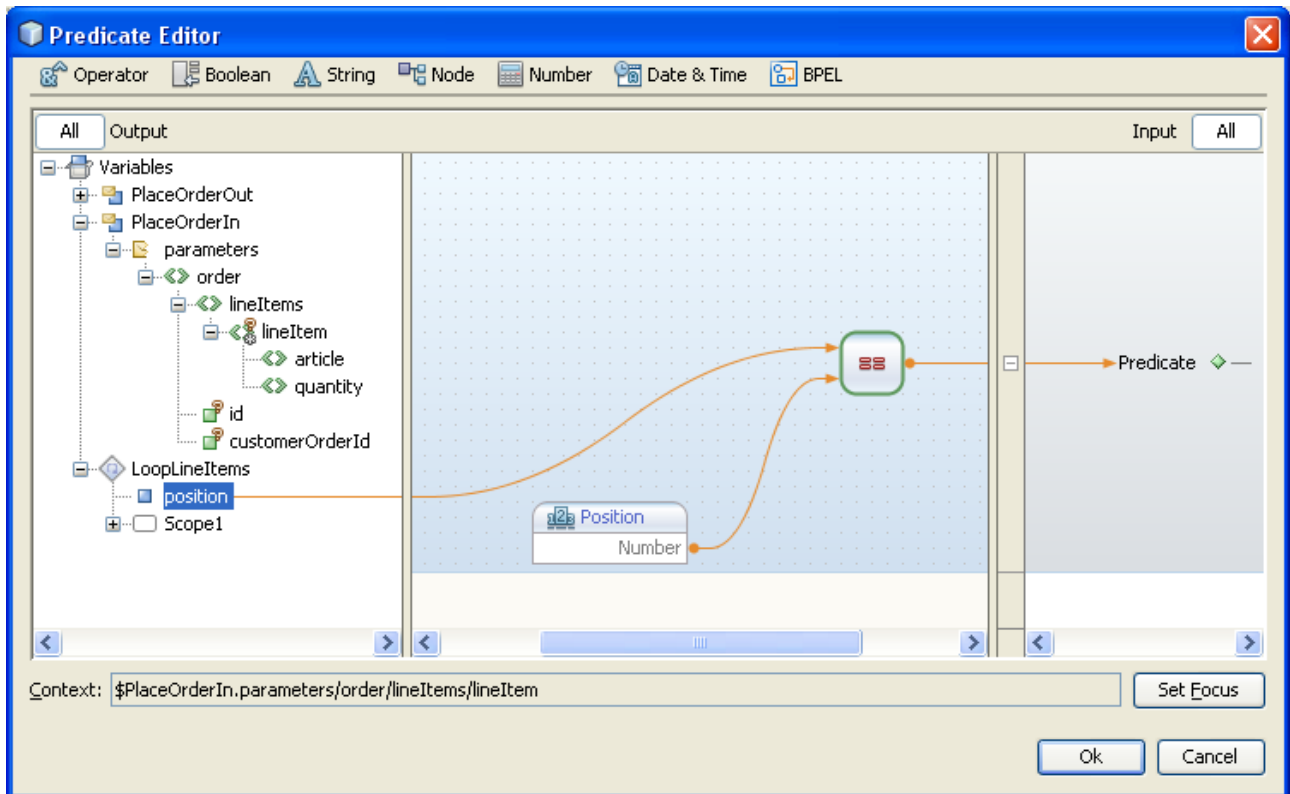


Abbildung 43: der Predicate Editor

12. Übergeben Sie die Werte von *article* und *quantity* der *PlaceOrderIn* Variablen an die *PlaceOrderIn1* Variable.

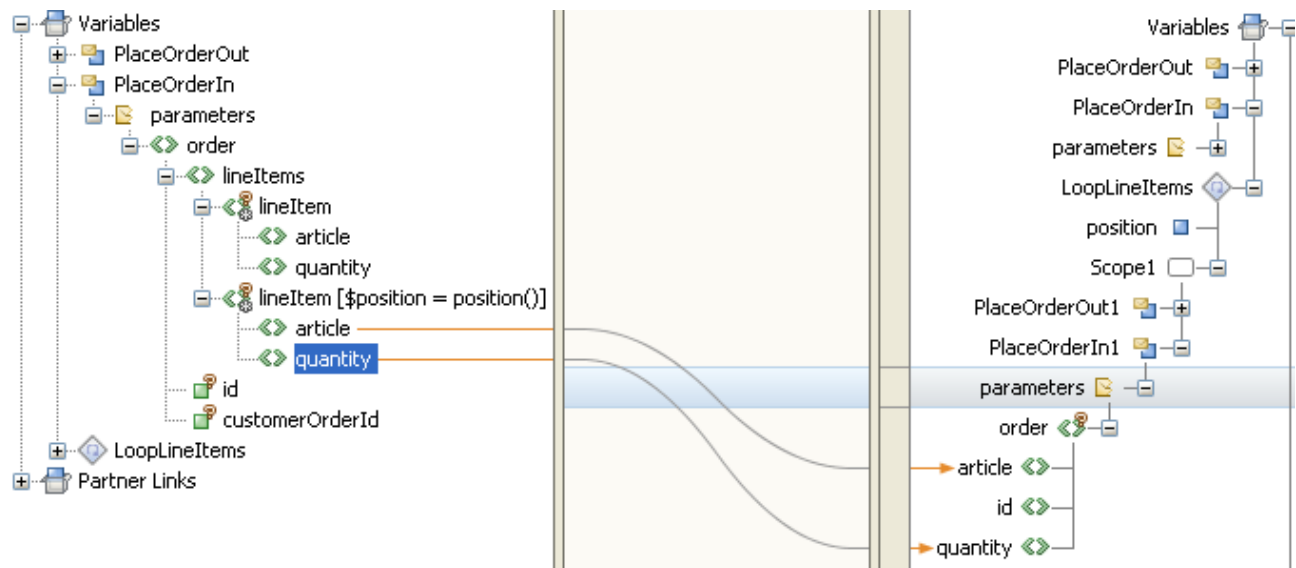


Abbildung 44: Übergeben der Werte

Übung: Debuggen

1. Führen Sie ein Deployment des Service durch.
2. Klicken Sie rechts auf das Element *Scope1* und wählen Sie *Toggle Breakpoint*.
3. Klicken Sie im Projekt *SlaughterhouseApp* rechts auf den Test und wählen Sie *Debug*.
4. Wechseln Sie zu der Ansicht *Watches*, klicken Sie rechts und wählen Sie *New Watch...*
5. Lassen Sie sich die Werte für *position* und *PlaceOrderIn1* ausgeben.
6. Wählen Sie den Debug Vorgang *Step Into* und betrachten Sie schrittweise die Änderungen der Variablen sowie die Vorgänge des Prozesses.
7. Erweitern Sie das Element *lineItems* der Datei *input.xml* des Tests um 2 weitere Bestellungen:

```
<ord:lineItems>
  <!--Zero or more repetitions:-->
  <ord:lineItem>
    <ord:article>0</ord:article>
    <ord:quantity>50</ord:quantity>
  </ord:lineItem>
  <ord:lineItem>
    <ord:article>1</ord:article>
    <ord:quantity>70</ord:quantity>
  </ord:lineItem>
  <ord:lineItem>
    <ord:article>2</ord:article>
    <ord:quantity>30</ord:quantity>
  </ord:lineItem>
</ord:lineItems>
```

8. Führen Sie den Debug Vorgang erneut aus.

Übung: Parallele Verarbeitung der Bestellungen

1. Führen Sie einen Rechtsklick auf die ForEach Aktivität *LoopLineItems* aus und wählen Sie *Go To/Source...*
2. Setzen Sie den Wert des Attributes *parallel* auf *yes*.

```
<forEach name="LoopLineItems" parallel="yes"
  counterName="position">
```

3. Wechseln Sie zur grafischen Ansicht und betrachten Sie die Warnung.

Übung: FaultHandling

1. Betrachten Sie im Projekt *SlaughterhouseService* die Klasse *PagingService*.
2. Lassen Sie sich das WSDL Dokument des Services anzeigen und kopieren Sie deren Adresse.
3. Fügen Sie dem Projekt *SlaughterhouseProcess* dieses Dokument als neues externes WSDL Dokument hinzu.
4. Erstellen Sie einen Partner Link *PagingServicePL*, der auf den Paging Service verweist, verwenden Sie *PagingServicePLT* für den Partner Link Type Namen.
5. Fügen Sie dem Element *Scope1* einen neuen *Fault Handler* hinzu.
6. Fügen Sie dem Fault Handler eine *Catch All* Aktivität hinzu.
7. Lassen Sie im Fehlerfall den Paging Service aufrufen und übergeben Sie diesem mit Hilfe einer Variablen den String "T-Bone Steaks wurden bestellt, konnten aber nicht gefertigt werden".
8. Bestellen Sie T-Bone Steaks (Artikel-ID: 3) und betrachten Sie die Ausgabe in der Glassfish Konsole.

Zusatzübung: Melden Sie den Artikel, welcher nicht gefertigt werden konnte.

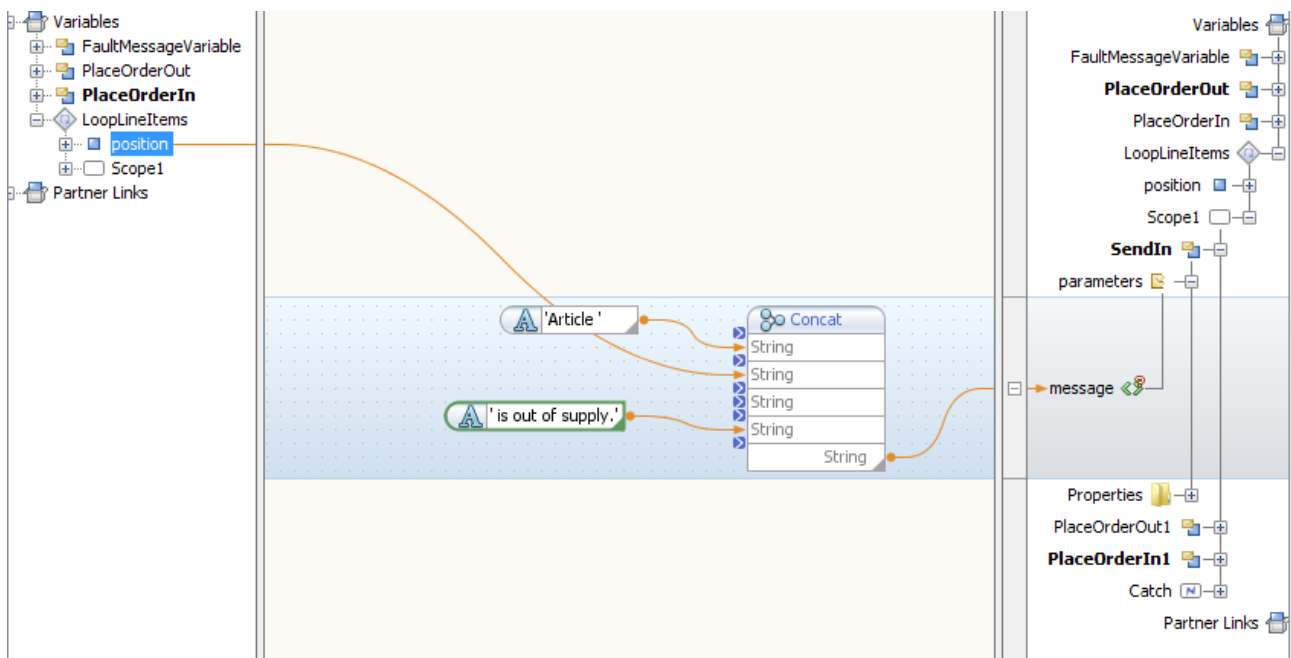


Abbildung 45: XPath für Fehlermeldung

7. Compensation

Übung: Kompensation

1. Entfernen Sie den Fault Handler und setzen Sie ein weiteres Scope Element um *Scope1*.
2. Fügen Sie dem *Scope1* einen Compensation Handler hinzu.
3. Fügen Sie dem neuen Handler ein Assign und ein darauf folgendes Invoke Element hinzu.
4. Konfigurieren Sie die neuen Elemente so, dass die *Cancel* Operation des *ProductionService* aufgerufen wird.
5. Testen Sie das Projekt und betrachten Sie das Ergebnis in der Glasfish Konsole.

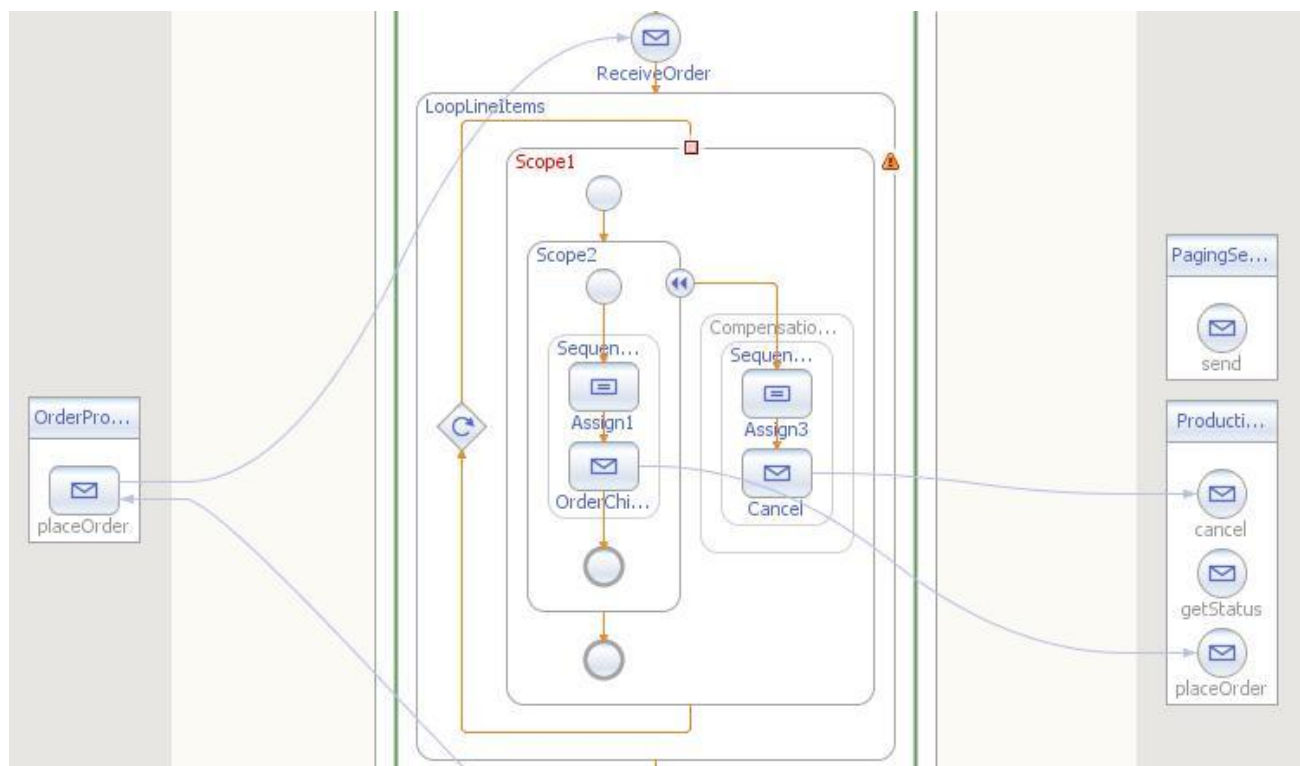


Abbildung 46: Compensation Handler

Der Code könnte wie folgt aussehen:

```
<compensationHandlers>
  <sequence name="Sequence4">
    <assign name="Assign3">
      <copy>
        <from>$PlaceOrderIn.parameters/ns0:order/ns0:lineItems/ns
          0:lineItem[position() = $position]/ns0:quantity</from>
        <to>$cancelIn.parameters/orderId</to>
      </copy>
    </assign>
    <invoke name="PagingService" partnerLink="PagingServicePL"
      operation="send" portType="ns2:PagingService"
      inputVariable="SendIn"/>
    </sequence>
  </compensationHandlers>
```

8. Correlation

Vorbereitung: Process erstellen

1. Importieren Sie das *EventProcess* Projekt vom Kursdisk in NetBeans.
2. Erstellen Sie ein entsprechend benanntes Composite Application Projekt für *EventProcess*.
3. Erstellen Sie einen Test Case für das Projekt und führen Sie ihn aus.

Frage: Was macht dieser Process?

Übung: Message Property Ticket anlegen

1. Betrachten Sie das *approve.wsdl* in *EventProcess* Projekt mit dem WSDL Editor.
2. Klicken Sie mit der rechten Maustaste auf *Extensibility Elements*, fügen Sie eine Property zu dem Dokument hinzu und konfigurieren Sie diese wie folgt.

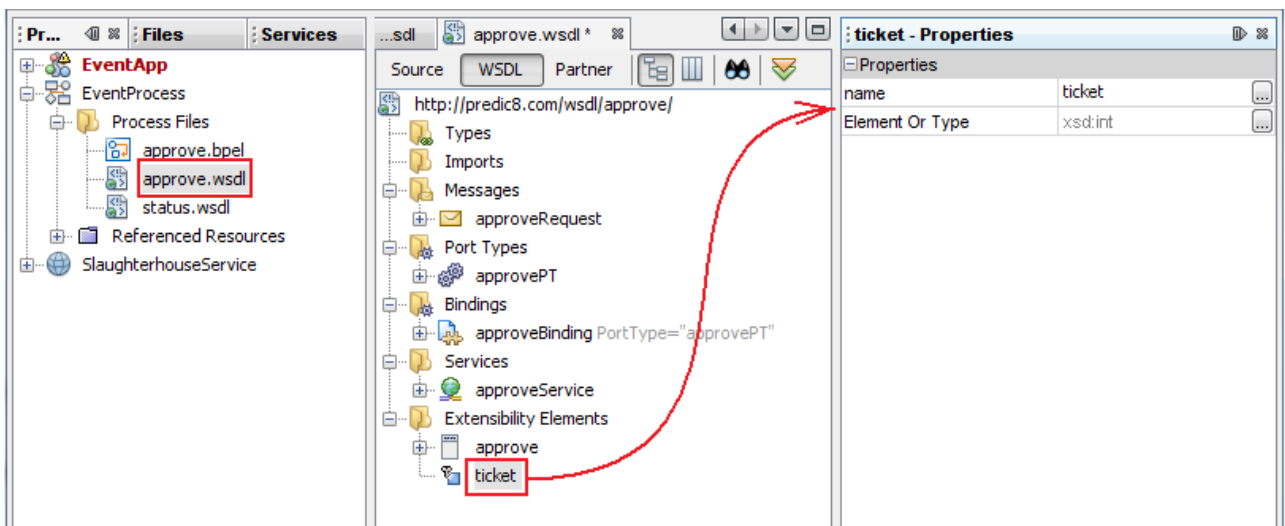


Abbildung 47: Property einfügen

3. Fügen Sie nun ein Property Alias ein und konfigurieren Sie ihn wie folgt, so dass er auf die eben erstellte Property zeigt.

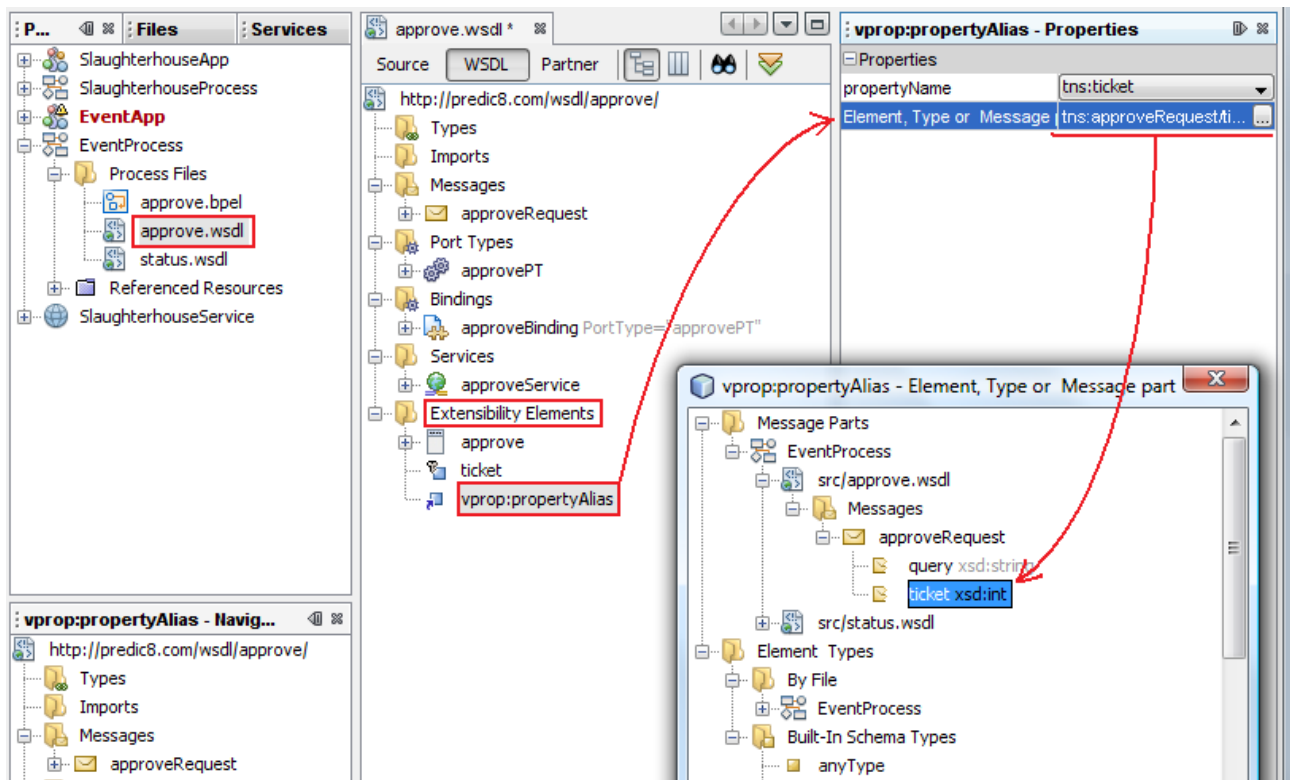


Abbildung 48: Property Alias einfügen

4. Wiederholen Sie Schritte eins bis drei auch für *status.wsdl*.

9. Event Handling

Übung: Correlation Set und Event Handler

1. Nun müssen die eben erstellten Properties in den Bpel Process eingebaut werden. Fügen Sie dafür wie im folgenden Screenshot ein Correlation Set zu dem Prozess hinzu.

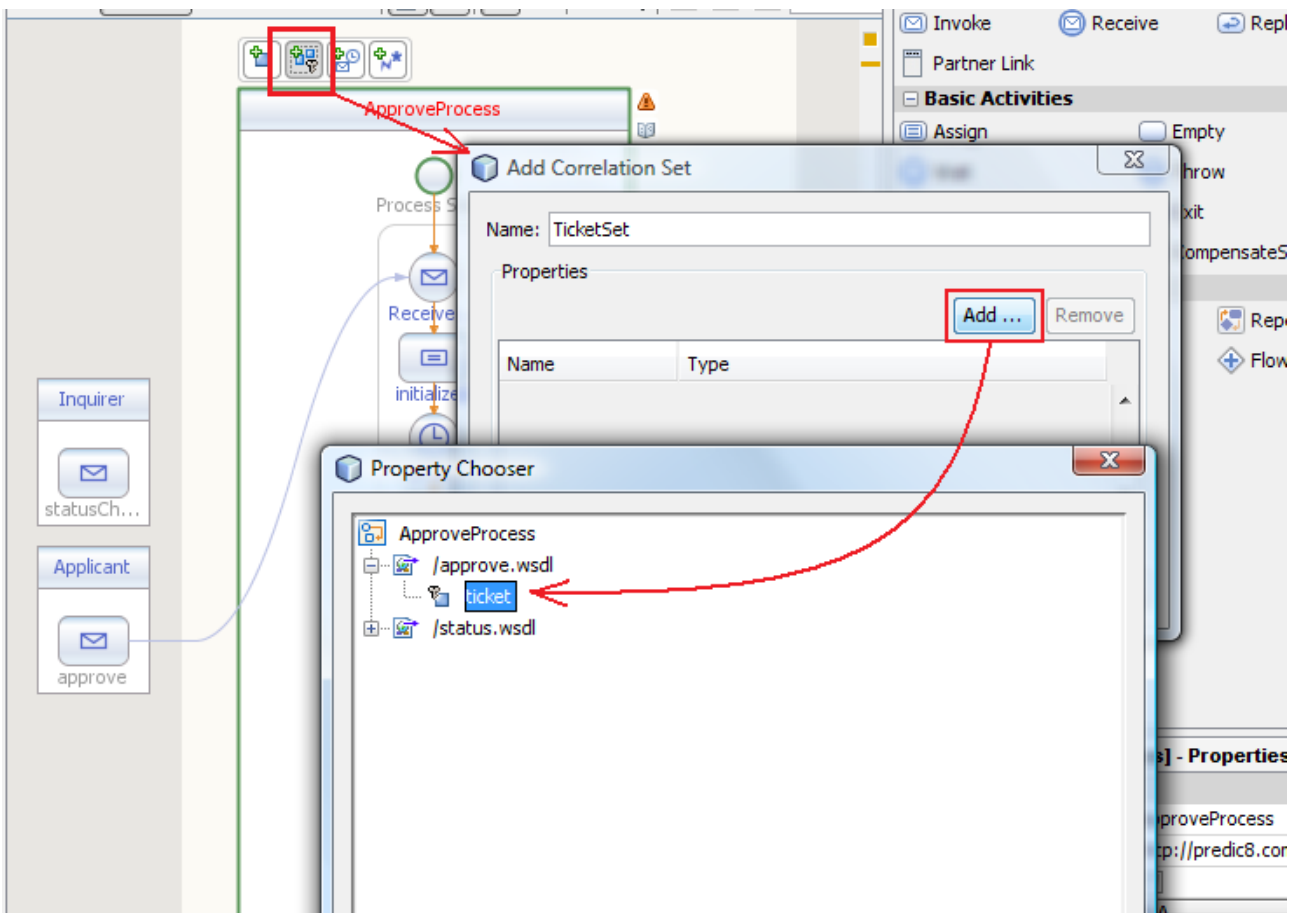
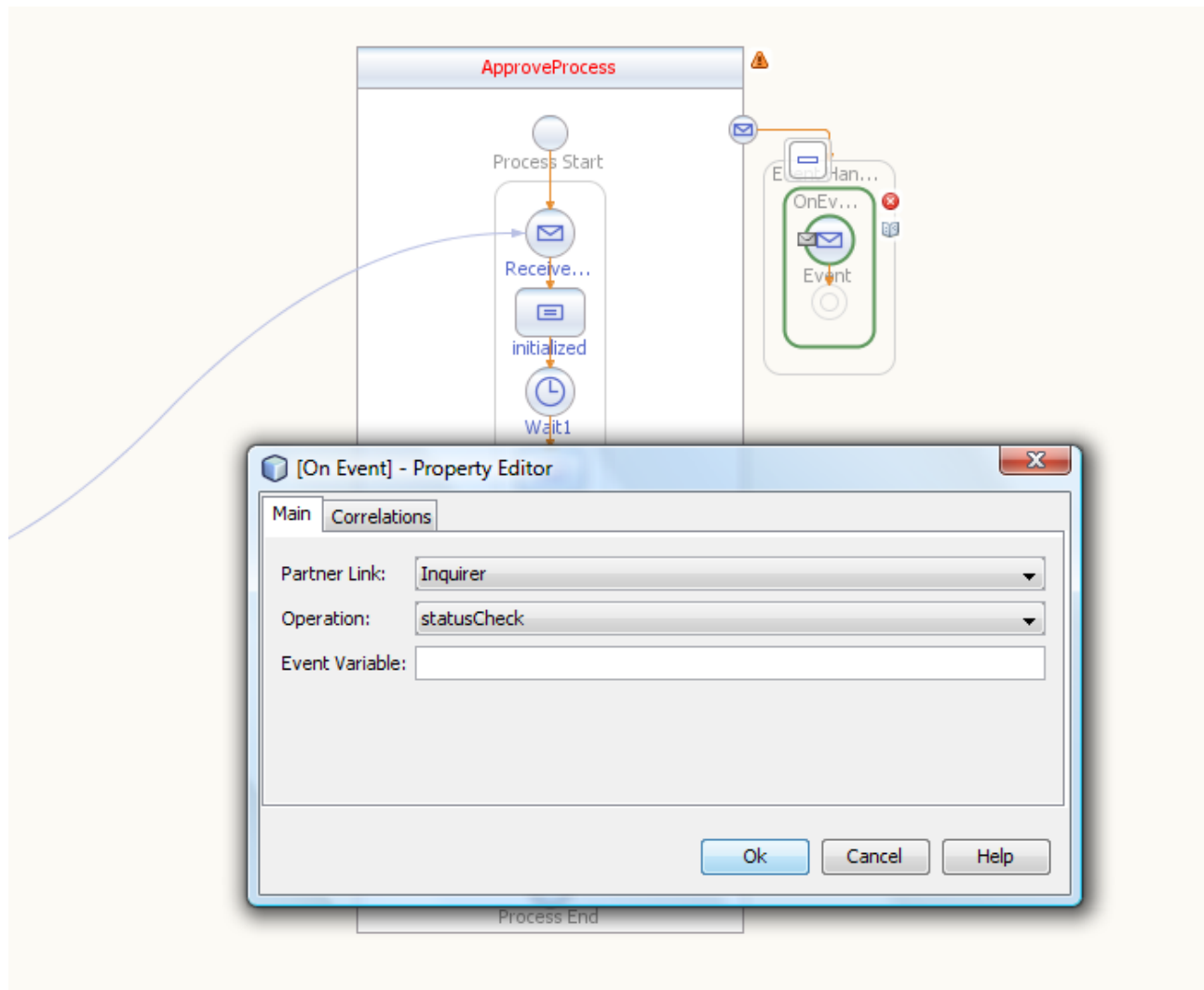


Abbildung 49: Erstellen eines Correlation Sets in Bpel Prozess

2. Fügen Sie dem Process einen Event Handler hinzu, indem Sie auf wieder auf den Prozess Klicken und aus der oberen Leiste *onEvent* auswählen.
3. Fügen Sie dem Event Handler einen weiteren *onEvent* hinzu.

4. Konfigurieren Sie das *onEvent* Element mit einem Doppelklick so, dass es von *Inquirer* aufgerufen wird. Eine Event Variable wird nicht benötigt.



5. Wechseln Sie zum *Correlations* Tab und fügen Sie das erstellte Correlation Set *TicketSet* hinzu, initialisieren sie es aber nicht.
6. Fügen Sie nun das *TicketSet* dem Recieve Element hinzu und ändern Sie den Wert für *initiate* zu *yes*.

7. Fügen Sie dem Event Handler eine Assign Aktivität hinzu und benennen Sie sie in getStatus um.

Bemerkung: Ein Event Handler darf nur ein Scope als Child Element besitzen. Es wird also automatisch ein Scope um das Assign gesetzt.

8. Fügen Sie in Scope1 eine neue Variable wie folgt hinzu.

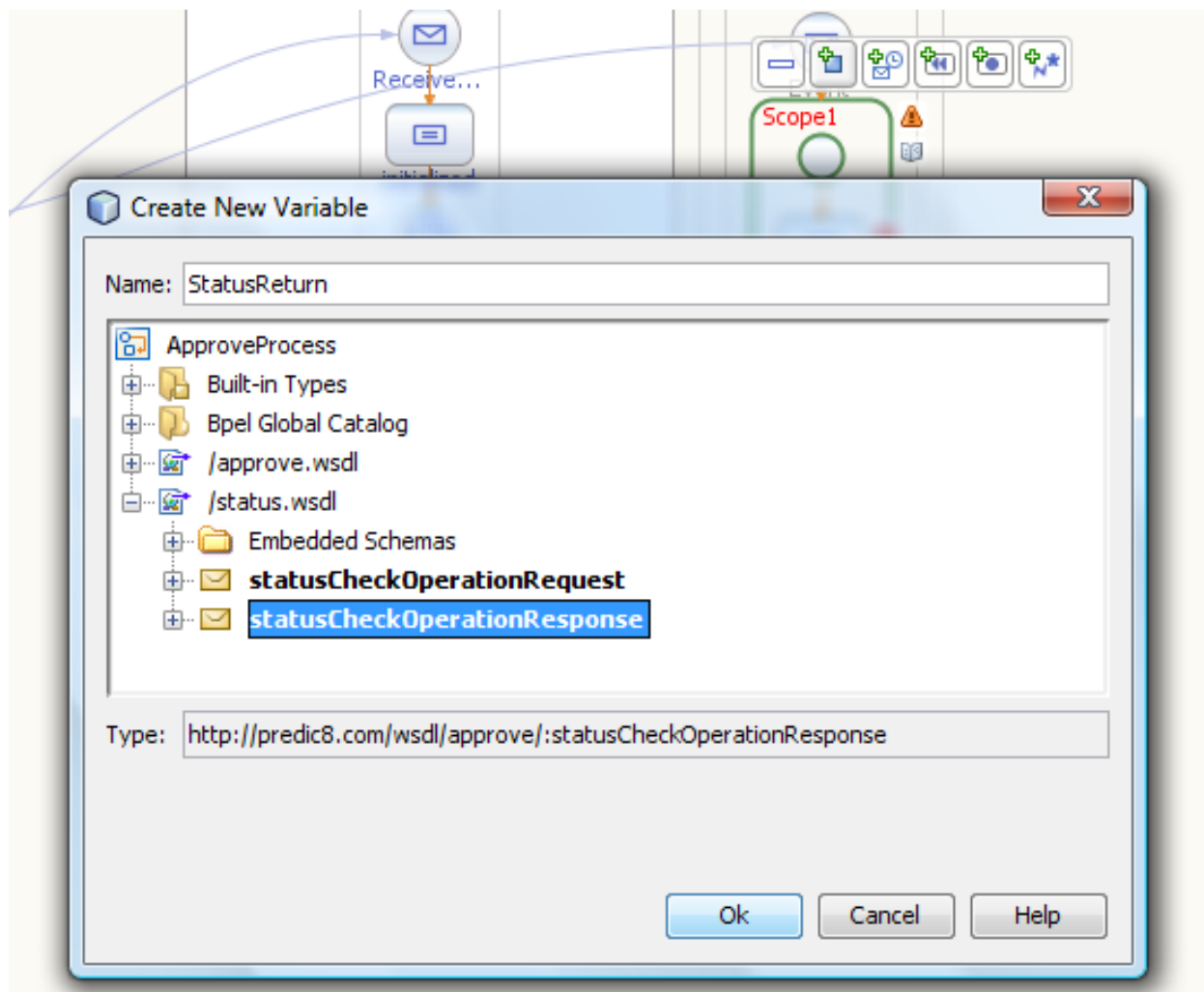


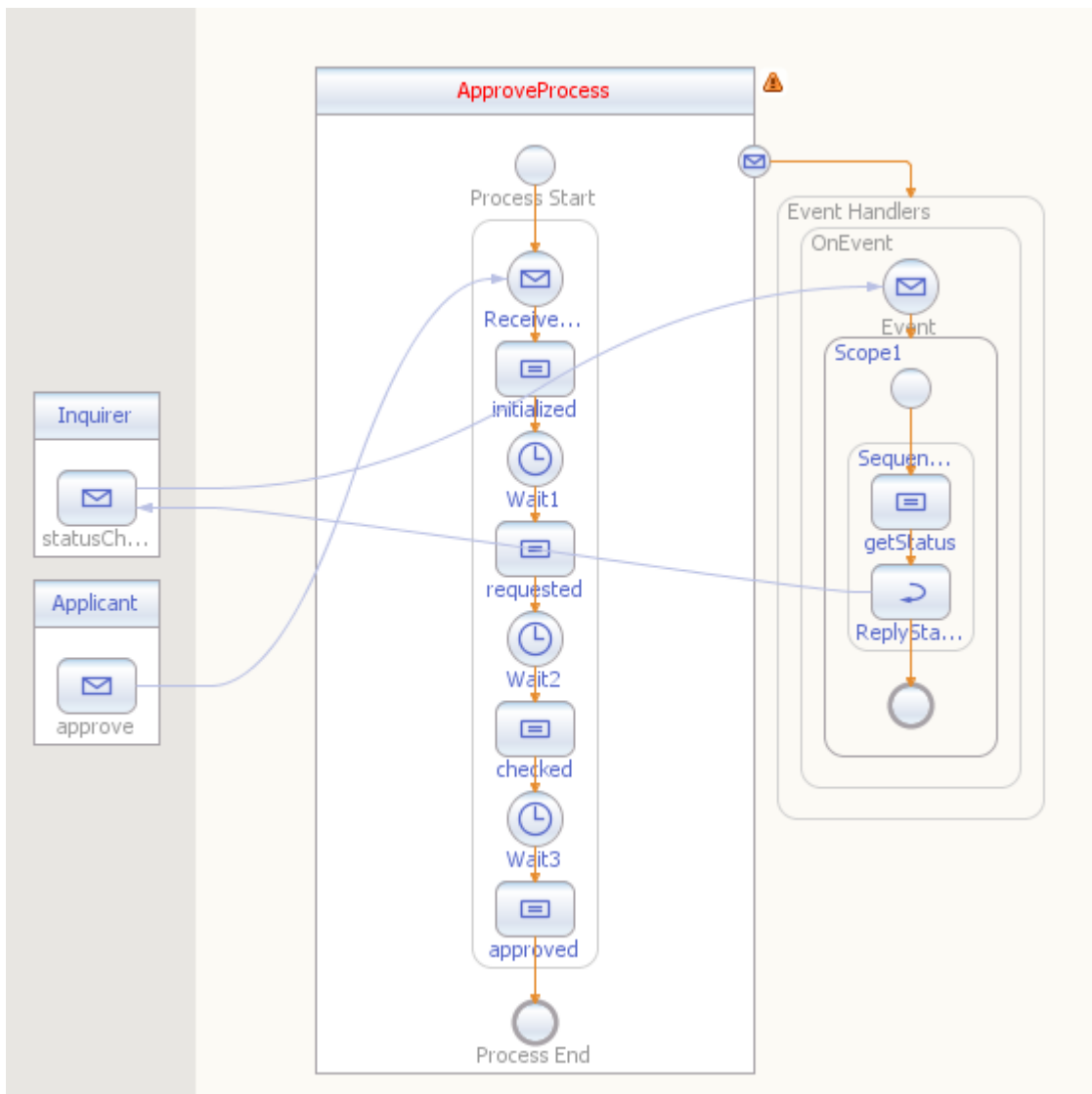
Abbildung 50: Erstellen einer neuen Variable in Scope1

9. Gehen Sie in die Mapper-Ansicht von *getStatus* und weisen Sie der *StatusReturn* Variable den Wert von *state* zu.



10. Fügen Sie nach dem Assign eine Reply Aktivität hinzu und benennen Sie sie in *ReplyStatus* um.
11. Konfigurieren Sie *ReplyStatus* wie folgt.

So sollte der Prozess nun aussehen:



Übung: Tests ausführen

1. Erstellen Sie einen neuen Test Case, der die Methode *statusCheck* von *status.wsdl* ausführt.
2. Achten Sie darauf, dass beide Input Messages den selben Wert für *ticket* besitzen.

Frage: Wie soll man die Test sinnvoll ausführen?

Frage: Was erwarten Sie von den Tests?

10. Pick

Übung: WSDL um eine Operation erweitern

Mit einem Aufruf von der neuen Operation *agree* soll ein Vorgesetzter einen Vorgang genehmigen können.

1. Importieren Sie das Projekt *PickProcess* aus dem Kursdisk-Ordner.
2. Öffnen Sie das *approve.wSDL* Dokument mit dem WSDL-Editor.
3. Fügen Sie dem Dokument eine neuen Operation wie folgt hinzu.

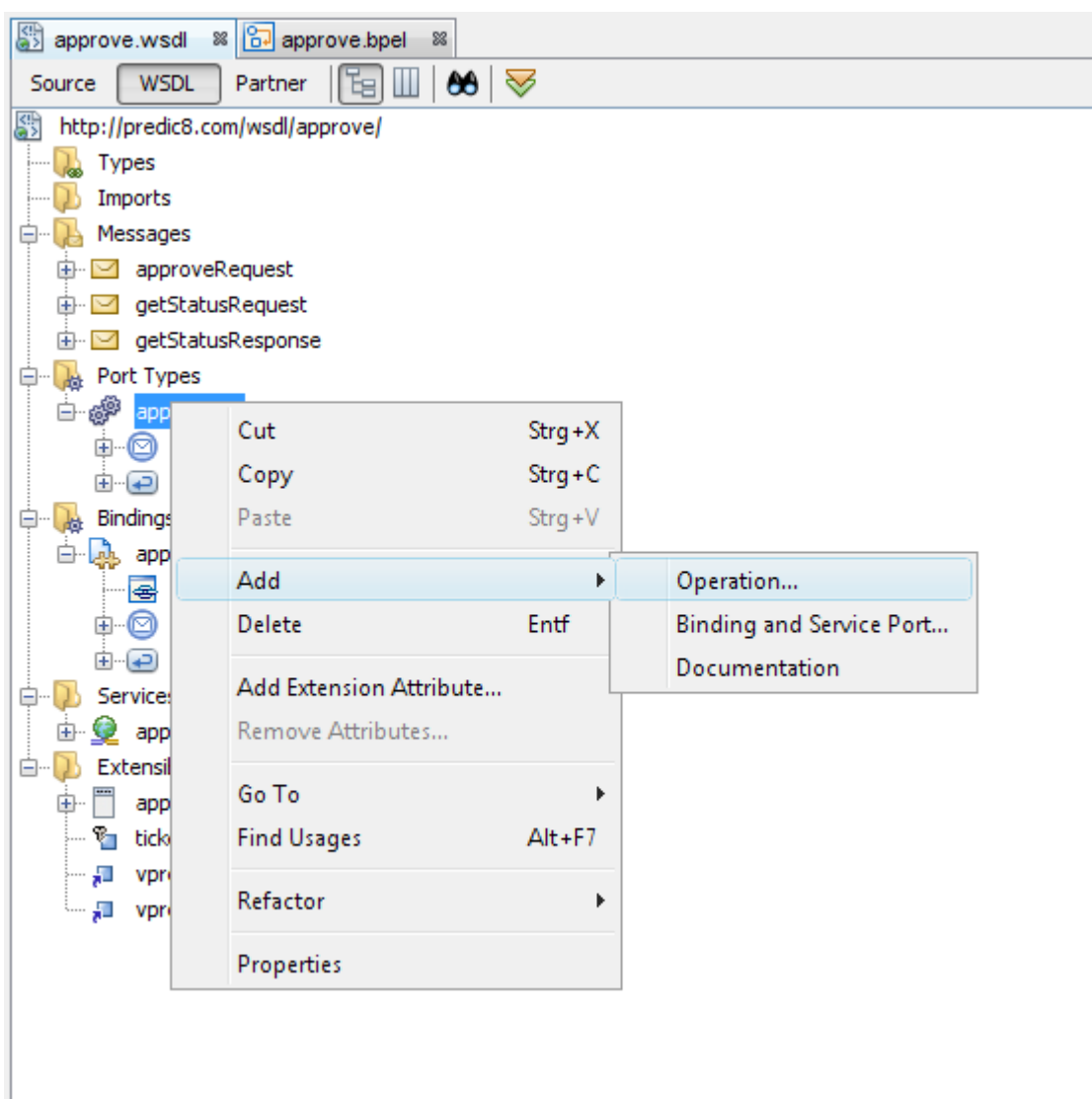
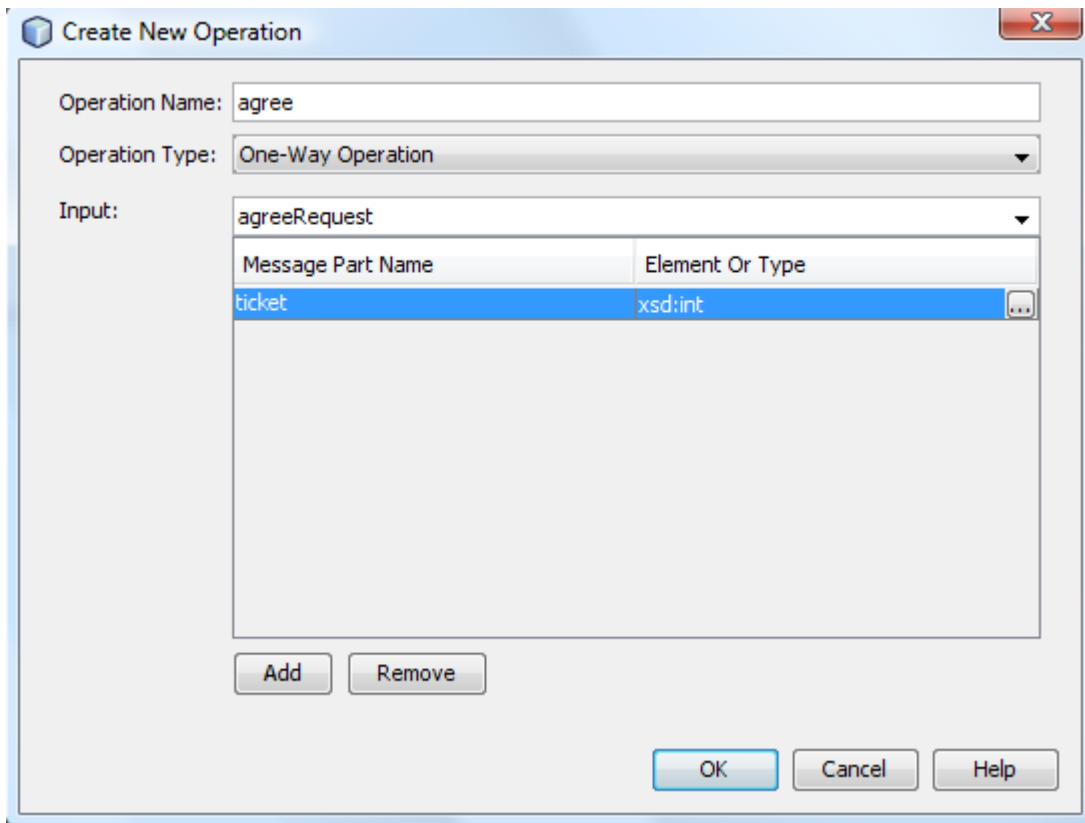


Abbildung 51: Neue Operation hinzufügen

4. Konfigurieren Sie die Operation wie folgt:



Create New Operation

Operation Name:

Operation Type:

Input:

Message Part Name	Element Or Type
ticket	xsd:int

Abbildung 52: Konfigurieren des Binding

5. Fügen Sie nun unter *Bindings/approveBinding* eine neue Binding Operation hinzu.

6. Betrachten Sie nun die erzeugte Binding Operation *agree* und die Input Variable *input3*. Fügen Sie dort einen SOAP Body ein und konfigurieren Sie ihn wie in der folgenden Abbildung.

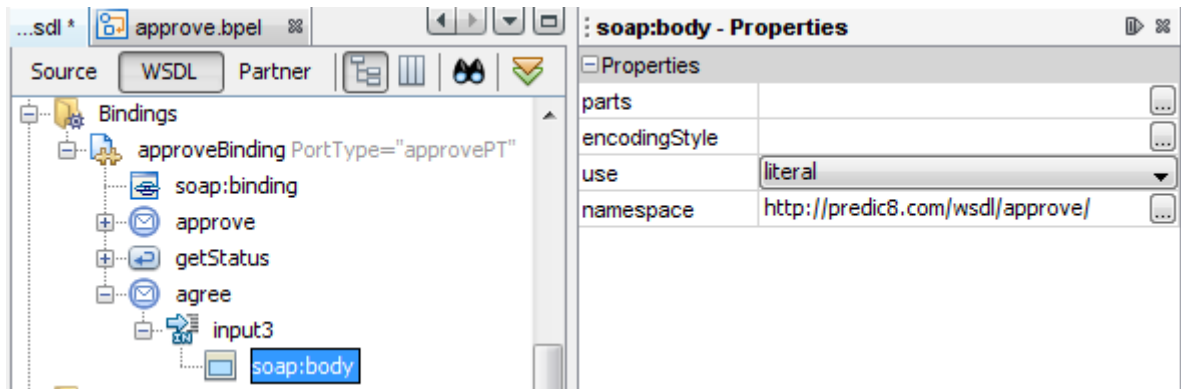


Abbildung 53: SOAP Body Konfigurieren

7. Fügen Sie jetzt ein neues Property Alias zu dem WSDL-Dokument hinzu und weisen Sie ihm die eben erstellte Variable *ticket* zu.

Übung: Bpel Prozess anpassen

1. Betrachten Sie den Process *approve.bpel*. Setzen Sie nach dem *initialized [Assign]* eine Pick Aktivität
2. Fügen Sie der Pick Aktivität ein neues **On Alarm** von der oberen Leiste hinzu.
3. Konfigurieren Sie den Timer für eine Dauer von einer Minute.
4. Setzen Sie ein neues Assign hinter dem Timer und benennen Sie es in *Declined* um
5. Konfigurieren Sie das Assign in der Mapper-Ansicht wie folgt:

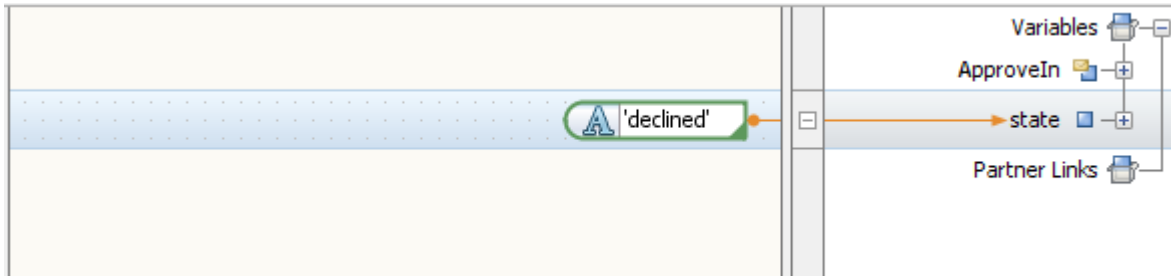


Abbildung 54: Declined[Assign] konfigurieren

6. Setzen Sie ein zweites Assign hinter dem *MessageHandler*, benennen Sie es in *Approved* um und konfigurieren Sie es entsprechend zu *Declined*, diesmal aber mit "approved" als Wert.
7. Konfigurieren Sie den *MessageHandler* mit einem Doppelklick, so dass er von *agree* aufgerufen werden kann.

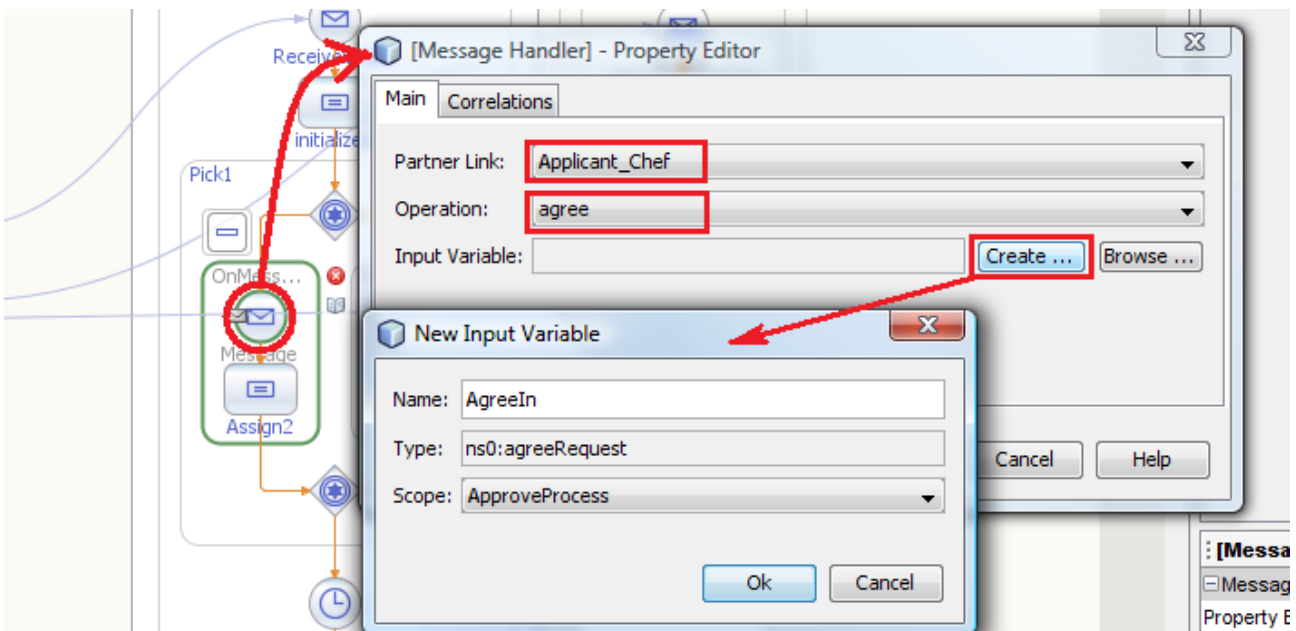


Abbildung 55: OnMessage Konfigurieren

8. Wechseln Sie auf das *Correlations* Tab und fügen Sie *TicketSet* zu der Liste der CorellationSets hinzu.
9. Klicken Sie schließlich auf OK.

So Sollte der Prozess nun aussehen:

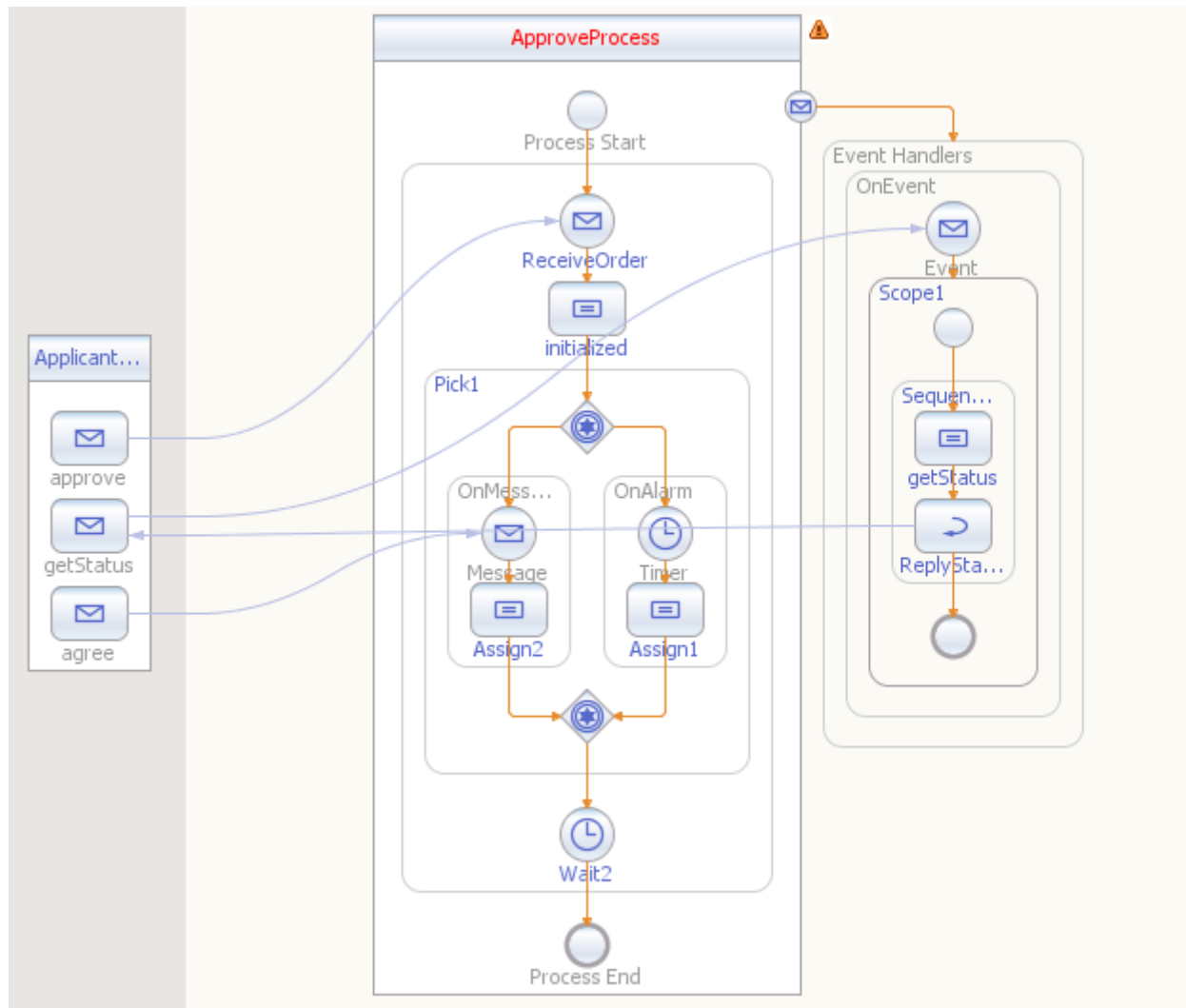


Abbildung 56: PickProcess

Übung: Erstellen Sie sinnvolle TestCases für den Process und testen Sie ihn.

Frage: Wie kann man von dem eingebauten CorrelationSet Gebrauch machen?

11. Business Activity Monitoring

Übung: Monitoring aktivieren

1. Rufen Sie die Eigenschaften der *sun-bpel-engine* auf.
2. Aktivieren Sie unter *Configuration* die Kästchen *Monitoring Enabled* und *Monitoring Variable Enabled*.

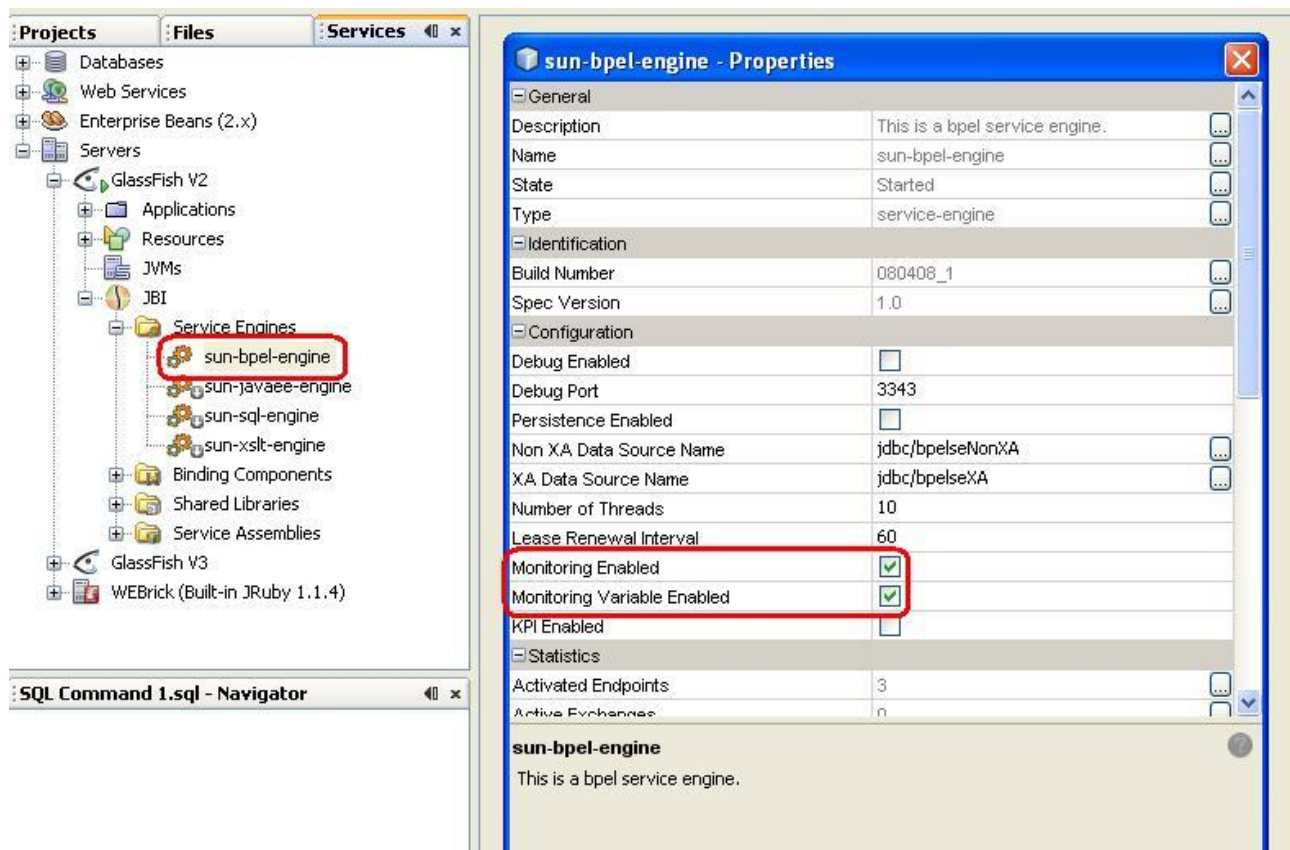


Abbildung 57: Monitoring aktivieren

3. Deployen Sie den Prozess erneut.

Übung: Monitoring über command line

Hinweis: Das benötigte Programm finden Sie auf der Kursdisk und im Web unter:

<http://wiki.open-esb.java.net/Wiki.jsp?page=BPELMonitor>

1. Führen sie ein Deployment für einen Prozess in Netbeans aus und lassen Sie ein Debugg laufen.
2. Öffnen Sie den Ordner *bpelMonitorTool\scripts* und starten Sie *runbpelmonitor.bat*
3. Lassen Sie sich die aktuellen Prozesse mit *b* anzeigen.
4. Speichern Sie die Ausgabe in eine Datei wie folgt:
`b >"c:\temp\output.txt"`
5. Mit *instid* können Sie die Prozesse explizit ansprechen. Finden Sie einen laufenden Prozess und terminieren Sie ihn.
6. Schauen Sie den Prozess in Netbeans an. Versuchen Sie den Debug fortzusetzen.

12. Troubleshooting

Wenn das Testen eines BPEL Prozess mehrmals fehl schlägt setzen Sie einen Breakpoint im unteren Bereich des Prozesses und versuchen Sie eine Ausführung des Debug-Modus.

12.1. Glassfish Probleme

Bei Problemen mit dem Glassfish Server V2:

1. Rechtsklicken Sie auf Glassfish und dann auf Remove.
2. Rechtsklicken auf die Server-Liste und wählen Sie New Server (Glassfish V2)
3. Wählen Sie einen neuen Admin-Ordner aus und erstellen Sie einen neuen Server.

Für laufende Projekte muss der Server eventuell neu angegeben werden.

12.2. Deployment Probleme

Wenn Sie ein Composite Application nicht mehr deployen können, löschen Sie den Bpel Process aus dem Projekt raus und importieren Sie ihn erneut.

Beachten Sie, dass die Dateinamen in Projekten, die gleichzeitig auf dem Server ausgeführt werden, unterschiedliche Namen haben müssen.