



## Übung 7

Dr. Malgorzata Mochol  
Freie Universität Berlin  
Institut für Informatik  
Netzbasierte Informationssysteme  
[mochol@inf.fu-berlin.de](mailto:mochol@inf.fu-berlin.de)

# Übung 7

- Musterlösungen zu Übungsblätter 5 & 6
- Web Service Komposition
- Musterfragen
- Klausur: organisatorische Hinweise



## Musterlösung des Übungsblattes 5

# Amazon E-Commerce Service (ECS)

- **Zwei Schnittstellen:**
  1. HTTP-GET (RESTful)
  2. SOAP über HTTP-POST
- **Welche SOAP-Version benutzt ECS?**
- **WSDL-Beschreibung von ECS:**

```
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/"  
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">  
</definitions>
```

⇒ **WSDL 1.1** ⇒ **benutzt SOAP 1.1.**

- **sieht man auch am SOAP-Namensraum**

## 1) HTTP-GET-Schnittstelle

- a) ECS-Aufruf: Bücher in amazon.de mit Titel „Harry Potter“
  
- b) ECS-Aufruf: Kunden-Reviews in amazon.de zu Buch mit ASIN 3551566666
  - Aufruf mit HTTP-GET
    - ⇒ Aufruf mit allen Parameter als URL kodieren
  - Resultat: XML

# 1a) Bücher mit Titel „Harry Potter“ Aufruf über HTTP-GET

Schlüssel

Wert


http://webservices.amazon.de/onca/xml  
?Service=AWSECommerceService  
&AWSAccessKeyId=[Access Key ID]  
&Operation=ItemSearch  
&SearchIndex=Books  
&Title=Harry+Potter

Search Index Name	US	UK	DE	JP	FR	CA
Apparel	✓					
Automotive	✓					
Baby	✓					
Beauty	✓					
Blended	✓	✓	✓	✓	✓	✓
Books	✓	✓	✓	✓	✓	✓
Classical	✓	✓	✓	✓	✓	✓
DigitalMusic	✓					
DVD	✓	✓	✓	✓	✓	✓

Quelle: „Amazon E-Commerce Service Developer Guide“

- Schlüssel und Werte sind case sensitive

# URL einfach in den Browser eingeben!



```
<?xml version="1.0" ?>
- <ItemSearchResponse xmlns="http://webservices.amazon.com/AWSECommerceService/2005-10-05">
- <OperationRequest>
  - <HTTPHeaders>
    <Header Name="UserAgent" Value="Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0; .NET CLR 2.0.50727; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729)" />
  </HTTPHeaders>
  <RequestId>f8740ca1-5013-4218-997a-fdf6f53ab83c</RequestId>
- <Arguments>
  <Argument Name="Operation" Value="ItemSearch" />
  <Argument Name="Service" Value="AWSECommerceService" />
  <Argument Name="AWSAccessKeyId" Value="15RKE0ES2QVVTVBT5B82" />
  <Argument Name="Title" Value="Harry Potter" />
  <Argument Name="SearchIndex" Value="Books" />
</Arguments>
  <RequestProcessingTime>0.0980250000000000</RequestProcessingTime>
</OperationRequest>
- <Items>
- <Request>
  <IsValid>True</IsValid>
  - <ItemSearchRequest>
    <Condition>New</Condition>
    <DeliveryMethod>Ship</DeliveryMethod>
    <MerchantId>Amazon</MerchantId>
    <ResponseGroup>Small</ResponseGroup>
    <SearchIndex>Books</SearchIndex>
    <Title>Harry Potter</Title>
  </ItemSearchRequest>
</Request>
  <TotalResults>467</TotalResults>
  <TotalPages>47</TotalPages>
+ <Item>
+ <Item>
```

```
- <Item>
  <ASIN>3551577773</ASIN>
  <DetailPageURL>http://www.amazon.de/Harry-Potter-Heiligt%C3%
    BCmer-Todes-Band/dp/3551577773%3FSubscriptionId%
    3D15RKE0ES2QVTVBT5B82%26tag%3Dws%26linkCode%
    3Dxm2%26camp%3D2025%26creative%3D165953%
    26creativeASIN%3D3551577773</DetailPageURL>
- <ItemAttributes>
  <Author>Joanne K. Rowling</Author>
  <Manufacturer>Carlsen</Manufacturer>
  <ProductGroup>Book</ProductGroup>
  <Title>Harry Potter und die Heiligtümer des Todes (Band 7)
    </Title>
</ItemAttributes>
</Item>
```

# und noch eine Fehlermeldung

```
<?xml version="1.0" ?>
- <ItemSearchResponse xmlns="http://webservices.amazon.com/AWSECommerceService/2005-10-05">
  - <OperationRequest>
    - <HTTPHeaders>
      <Header Name="UserAgent" Value="Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1; Trident/4.0; .NET CLR 2.0.50727; .NET CLR 3.0.4506.2152; .NET CLR 3.5.30729)" />
    </HTTPHeaders>
    <RequestId>5ebdf273-4182-4382-999e-26f5bf308f87</RequestId>
    <Arguments>
      <Argument Name="Operation" Value="ItemSearch" />
      <Argument Name="Service" Value="AWSECommerceService" />
      <Argument Name="AWSAccessKeyId" Value="15RKE0ES" />
      <Argument Name="Title" Value="Harry Potter" />
      <Argument Name="SearchIndex" Value="Books" />
    </Arguments>
    <RequestProcessingTime>0.0036910000000000</RequestProcessingTime>
  </OperationRequest>
  - <Items>
    - <Request>
      <IsValid>False</IsValid>
      - <ItemSearchRequest>
        <SearchIndex>Books</SearchIndex>
        <Title>Harry Potter</Title>
      </ItemSearchRequest>
      - <Errors>
        - <Error>
          <Code>AWS.InvalidParameterValue</Code>
          <Message>15RKE0ES is not a valid value for AWSAccessKeyId. Please change this value and retry your request.</Message>
        </Error>
      </Errors>
    </Request>
  </Items>
</ItemSearchResponse>
```

Falscher KeyID

konkreter Fehler

# 1b) Kunden-Reviews zu 3551566666 Aufruf über HTTP-GET

Schlüssel

```
http://webservices.amazon.de/onca/xml
  ?Service=AWSECommerceService
  &AWSAccessKeyId=...
  &Operation=ItemLookup
  &ItemId=3551566666
  &ResponseGroup=Reviews
  &ReviewPage=10
```

Wert

- Warum entspricht dies dem REST-Grundsatz?

- **REST-Architektur des WWW (Fielding 2000):**  
jede Web-Ressource soll eindeutig über eine URI identifiziert werden
- **Beispiel: online gebuchte Reise = Web-Ressource**
- gebuchte Reise sollte daher auch über eine URI eindeutig identifiziert werden

# REST oder nicht, das ist die Frage!

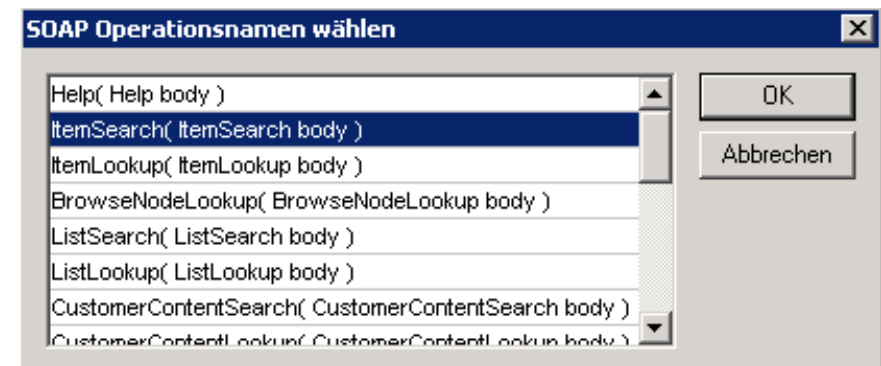
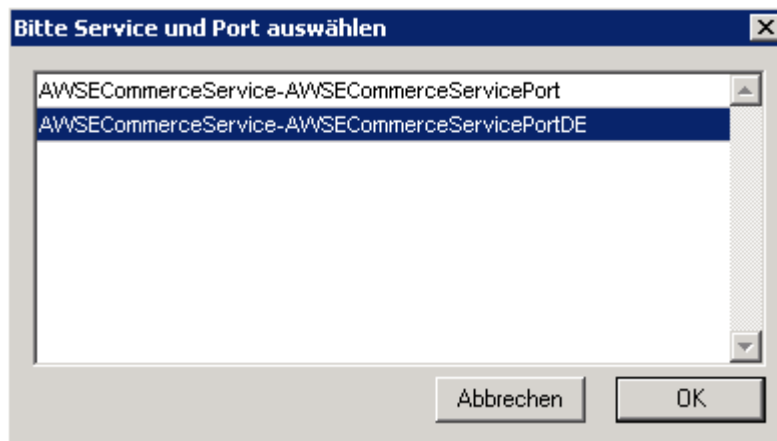
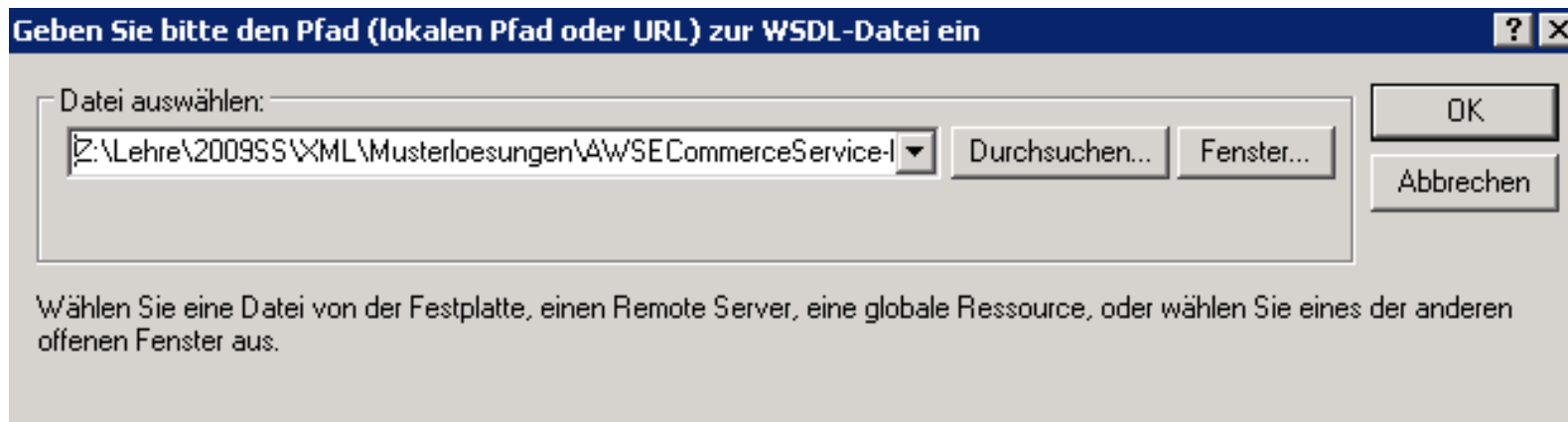
```
http://webservices.amazon.de/onca/xml  
    ?Service=AWSECommerceService  
    &AWSAccessKeyId=...  
    &Operation=ItemLookup  
    &ItemId=3551566666  
    &ResponseGroup=Reviews  
    &ReviewPage=10
```

- Kunden-Reviews zu 3551566666 = Web-Ressource = URL  
⇒ entspricht REST-Grundsatz

## 2) SOAP-Anfrage

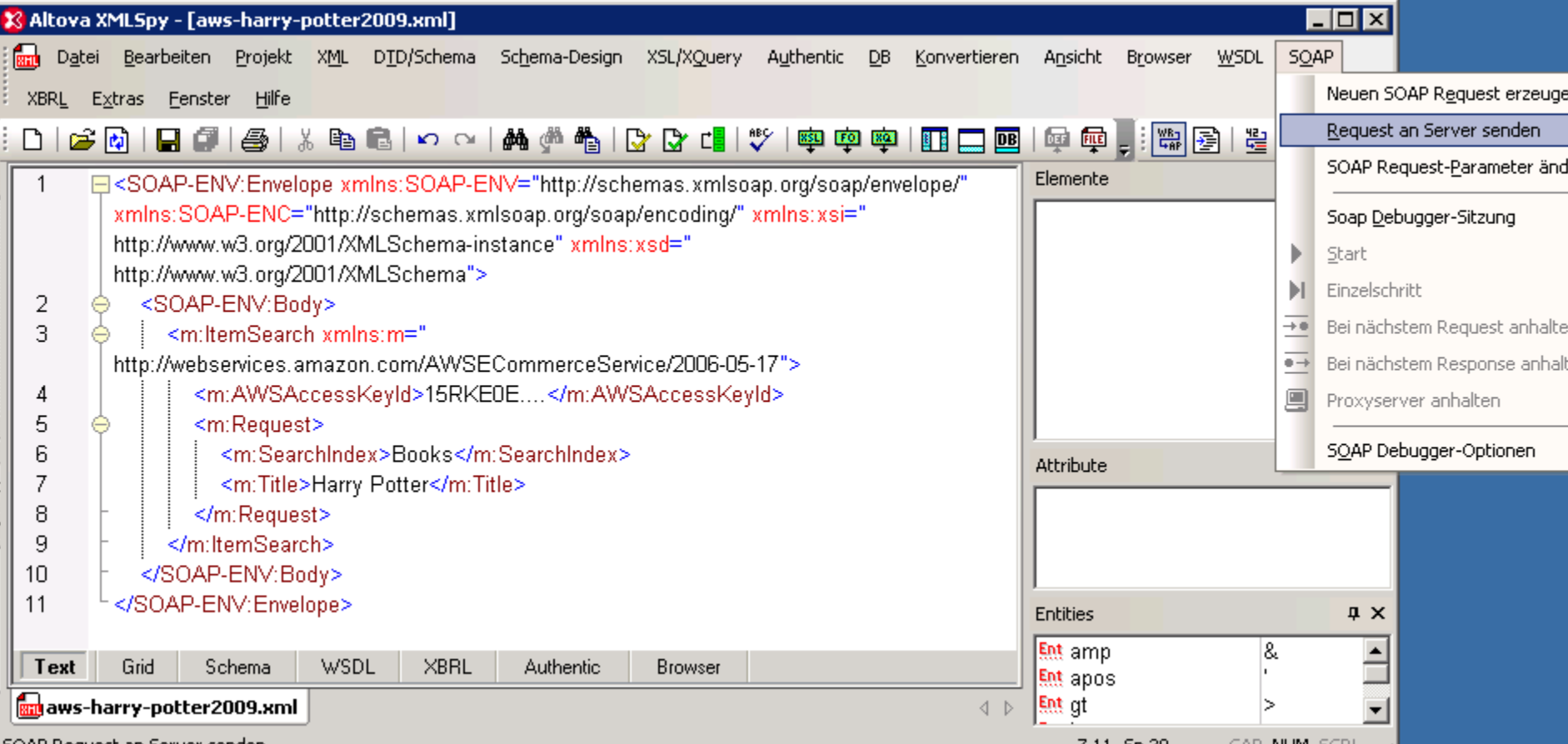
- a) ECS-Aufruf: Bücher in amazon.de mit Titel „Harry Potter“
- b) ECS-Aufruf: Kunden-Reviews in amazon.de zu Buch mit ASIN 3551566666
  - Aufruf mit HTTP-POST
  - Resultat: SOAP-Nachricht

# SOAP-Anfrage erzeugen



```
1  <SOAP-ENV:Envelope xmlns:SOAP-ENV="
    http://schemas.xmlsoap.org/soap/envelope/" xmlns:SOAP-ENC="
    http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="
    http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="
    http://www.w3.org/2001/XMLSchema">
2  <SOAP-ENV:Body>
3  <m:ItemSearch xmlns:m="
    http://webservices.amazon.com/AWSECommerceService/2006-05-17">
4      <m:MarketplaceDomain>String</m:MarketplaceDomain>
5      <m:AWSAccessKeyId>String</m:AWSAccessKeyId>
6      <m:SubscriptionId>String</m:SubscriptionId>
7      <m:AssociateTag>String</m:AssociateTag>
8      <m:XMLEscaping>String</m:XMLEscaping>
9      <m:Validate>String</m:Validate>
10     <m:Shared>
11         <m:Actor>String</m:Actor>
12         <m:Artist>String</m:Artist>
13         <m:Availability>String</m:Availability>
14         <m:AudienceRating>String</m:AudienceRating>
15         <m:Author>String</m:Author>
16         <m:Brand>String</m:Brand>
17         <m:BrowseNode>String</m:BrowseNode>
18         <m:City>String</m:City>
19         <m:Composer>String</m:Composer>
20         <m:Condition>String</m:Condition>
```

## 2a) Bücher mit Titel „Harry Potter“ SOAP-Anfrage editieren & request senden



The screenshot shows the Altova XMLSpy interface with the following XML content:

```
1 <SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
2   xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="
3   http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="
4   http://www.w3.org/2001/XMLSchema">
5   <SOAP-ENV:Body>
6     <m:ItemSearch xmlns:m="
7       http://webservices.amazon.com/AWSECommerceService/2006-05-17">
8       <m:AWSAccessKeyId>15RKE0E....</m:AWSAccessKeyId>
9       <m:Request>
10        <m:SearchIndex>Books</m:SearchIndex>
11        <m:Title>Harry Potter</m:Title>
12      </m:Request>
13    </m:ItemSearch>
14  </SOAP-ENV:Body>
15 </SOAP-ENV:Envelope>
```

The SOAP Debugger menu is open, showing the following options:

- Neuen SOAP Request erzeugen
- Request an Server senden**
- SOAP Request-Parameter ändern
- Soap Debugger-Sitzung
  - Start
  - Einzelanschritt
  - Bei nächstem Request anhalten
  - Bei nächstem Response anhalten
  - Proxyserver anhalten
- SOAP Debugger-Optionen

The status bar at the bottom indicates: SOAP Request an Server senden, Z 11, Sp 20, CAP NUM SCRL

# SOAP-Envelope (in Grid-View)

SOAP-ENV:Envelope	
<b>xmlns:SOAP-ENV</b>	http://schemas.xmlsoap.org/soap/envelope/
<b>xmlns:SOAP-ENC</b>	http://schemas.xmlsoap.org/soap/encoding/
<b>xmlns:xsi</b>	http://www.w3.org/2001/XMLSchema-instance
<b>xmlns:xsd</b>	http://www.w3.org/2001/XMLSchema
SOAP-ENV:Body	
<b>m:ItemSearch</b>	
<b>xmlns:m</b>	http://webservices.amazon.com/AWSECommerceService/2006-05-17
<b>m:AWSAccessK...</b>	15RKE0E...
<b>m:Request</b>	
<b>m:SearchIndex</b>	Books
<b>m:Title</b>	Harry Potter

# Fehlermeldung

```
2 <SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
3   <SOAP-ENV:Body>
4     <ItemSearchResponse xmlns="http://webservices.amazon.com/AWSECommerceService/2006-05-17">
5       <OperationRequest>
6         <HTTPHeaders>
7           <Header Name="UserAgent" Value="XML Spy"/>
8         </HTTPHeaders>
9         <RequestId>69e5e57f-7178-42ba-a8b3-2cbc8bd381b3</RequestId>
10        <Arguments>
11          <Argument Name="Service" Value="AWSECommerceService"/>
12        </Arguments>
13        <RequestProcessingTime>0.0043960000000000</RequestProcessingTime>
14      </OperationRequest>
15      <Items>
16        <Request>
17          <IsValid>False</IsValid>
18          <ItemSearchRequest>
19            <SearchIndex>Books</SearchIndex>
20            <Title>Harry Potter</Title>
21          </ItemSearchRequest>
22          <Errors>
23            <Error>
24              <Code>AWS.InvalidParameterValue</Code>
25              <Message>15RKEDE is not a valid value for AWSAccessKeyId. Please change this value and retry your
26            request.</Message>
27            </Error>
28          </Errors>
29        </Request>
30      </Items>
31    </ItemSearchResponse>
32  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

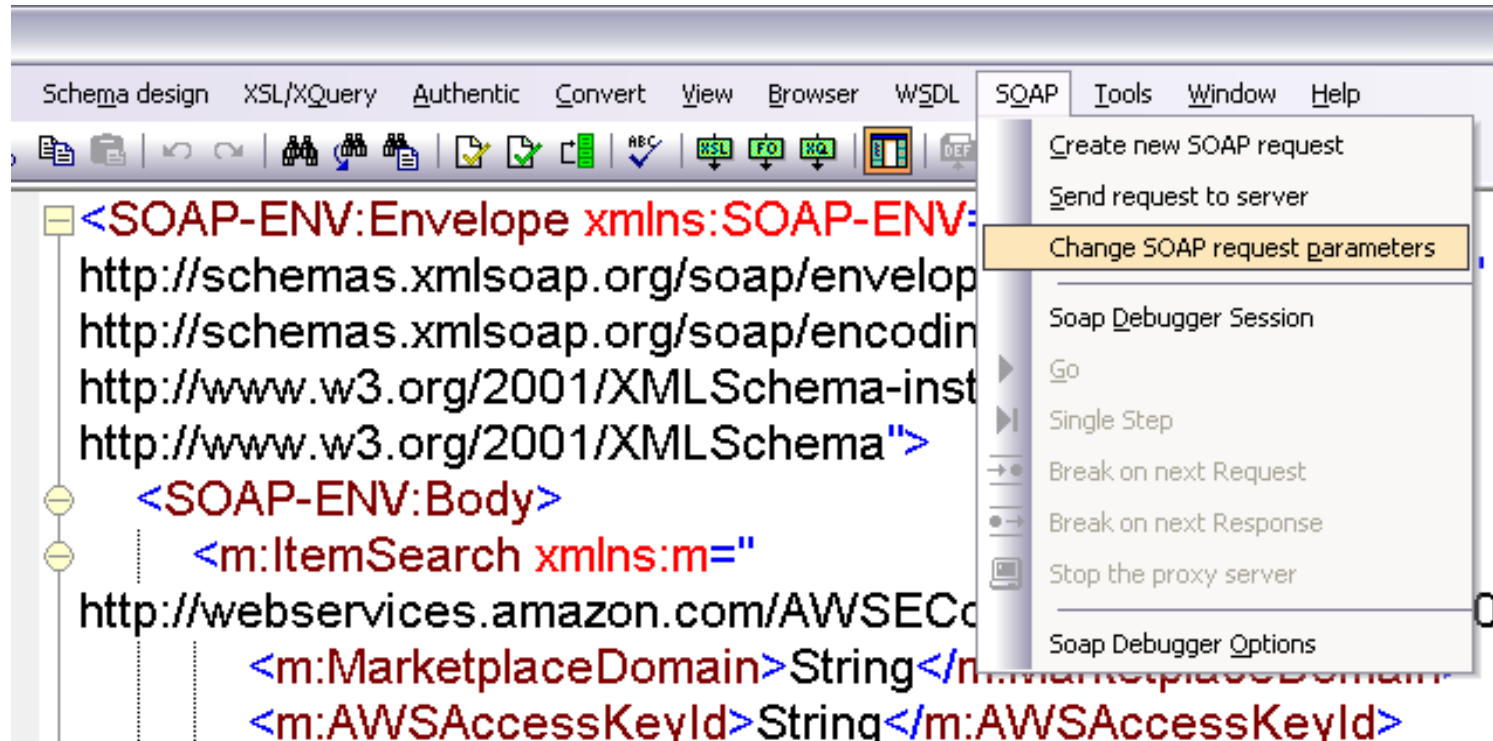
ungültige Anfrage (Request)

konkreter Fehler

# Die SOAP-Antwort: erster Treffer

```
15 <Items>
16   <Request>
17     <IsValid>True</IsValid>
18     <ItemSearchRequest>
19       <Condition>New</Condition>
20       <DeliveryMethod>Ship</DeliveryMethod>
21       <MerchantId>Amazon</MerchantId>
22       <ResponseGroup>Small</ResponseGroup>
23       <SearchIndex>Books</SearchIndex>
24       <Title>Harry Potter</Title>
25     </ItemSearchRequest>
26   </Request>
27   <TotalResults>467</TotalResults>
28   <TotalPages>47</TotalPages>
29   <Item>
30     <ASIN>3551577773</ASIN>
31     <DetailPageURL>
32       http://www.amazon.de/Harry-Potter-Heiligt%C3%BCmer-Todes-Band/dp/3551577773%3FSubscriptionId%3D15RKE0ES2QVWTVBT5B82%26tag%3Dws%26linkCode%3Dsp1%26camp%3D2025%26creative%3D165953%26creativeASIN%3D3551577773</DetailPageURL>
33     <ItemAttributes>
34       <Author>Joanne K. Rowling</Author>
35       <Manufacturer>Carlsen</Manufacturer>
36       <ProductGroup>Book</ProductGroup>
37       <Title>Harry Potter und die Heiligtümer des Todes (Band 7)</Title>
38     </ItemAttributes>
39   </Item>
```

# Endpunkt (URL) ändern



# Endpunkt (URL) ändern

**SOAP Request-Einstellungen** [X]

Connection Endpoint:

SOAPAction:

als SOAP+XML( Soap 1.2 ) senden

Änderung von **.de** zu **.com**

**SOAP Request-Einstellungen** [X]

Connection Endpoint:

SOAPAction:

als SOAP+XML( Soap 1.2 ) senden

# Die SOAP-Antwort (erster Treffer)

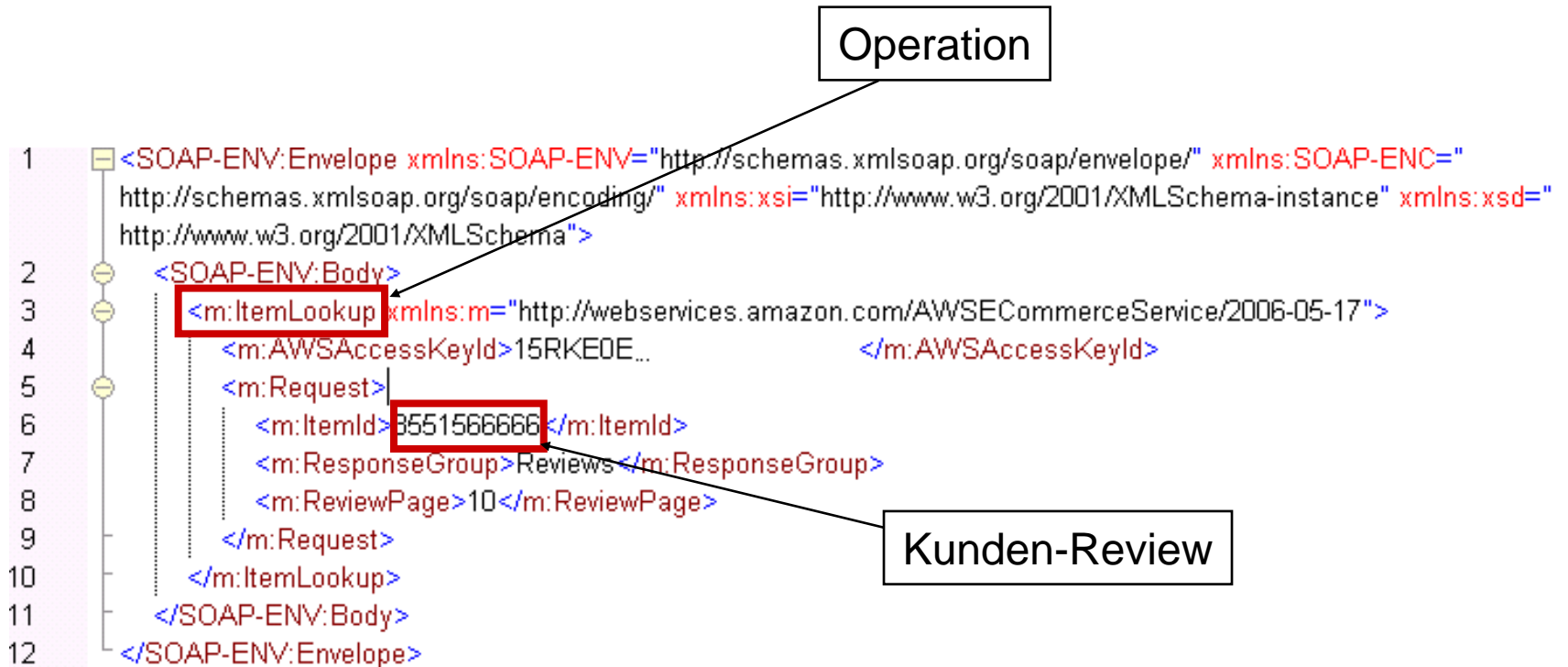
Nach Endpoint-Änderung  
(<http://soap.amazon.com>)

```
29 <Item>
30   <ASIN>0545139708</ASIN>
31   <DetailPageURL>
32     http://www.amazon.com/Harry-Potter-Deathly-Hallows-Rowling/dp/0545139708%3FSubscriptionId%3D15RKE0ES2QVVTVBT5B82%26tag%3Dws%26linkCode%3Dsp1%26camp%3D2025%26creative%3D165953%26creativeASIN%3D0545139708</DetailPageURL>
33   <ItemAttributes>
34     <Author>J.K. Rowling</Author>
35     <Manufacturer>Arthur A. Levine Books</Manufacturer>
36     <ProductGroup>Book</ProductGroup>
37     <Title>Harry Potter And The Deathly Hallows</Title>
38   </ItemAttributes>
</Item>
```

Vor Endpoint-Änderung  
(<http://soap.amazon.de>)

```
29 <Item>
30   <ASIN>3551577773</ASIN>
31   <DetailPageURL>
32     http://www.amazon.de/Harry-Potter-Heiligt%C3%BCmer-Todes-Band/dp/3551577773%3FSubscriptionId%3D15RKE0ES2QVVTVBT5B82%26tag%3Dws%26linkCode%3Dsp1%26camp%3D2025%26creative%3D165953%26creativeASIN%3D3551577773</DetailPageURL>
33   <ItemAttributes>
34     <Author>Joanne K. Rowling</Author>
35     <Manufacturer>Carlsen</Manufacturer>
36     <ProductGroup>Book</ProductGroup>
37     <Title>Harry Potter und die Heiligtümer des Todes (Band 7)</Title>
38   </ItemAttributes>
</Item>
```

## 2b) Kunden-Reviews zu 3551566666 SOAP-Anfrage



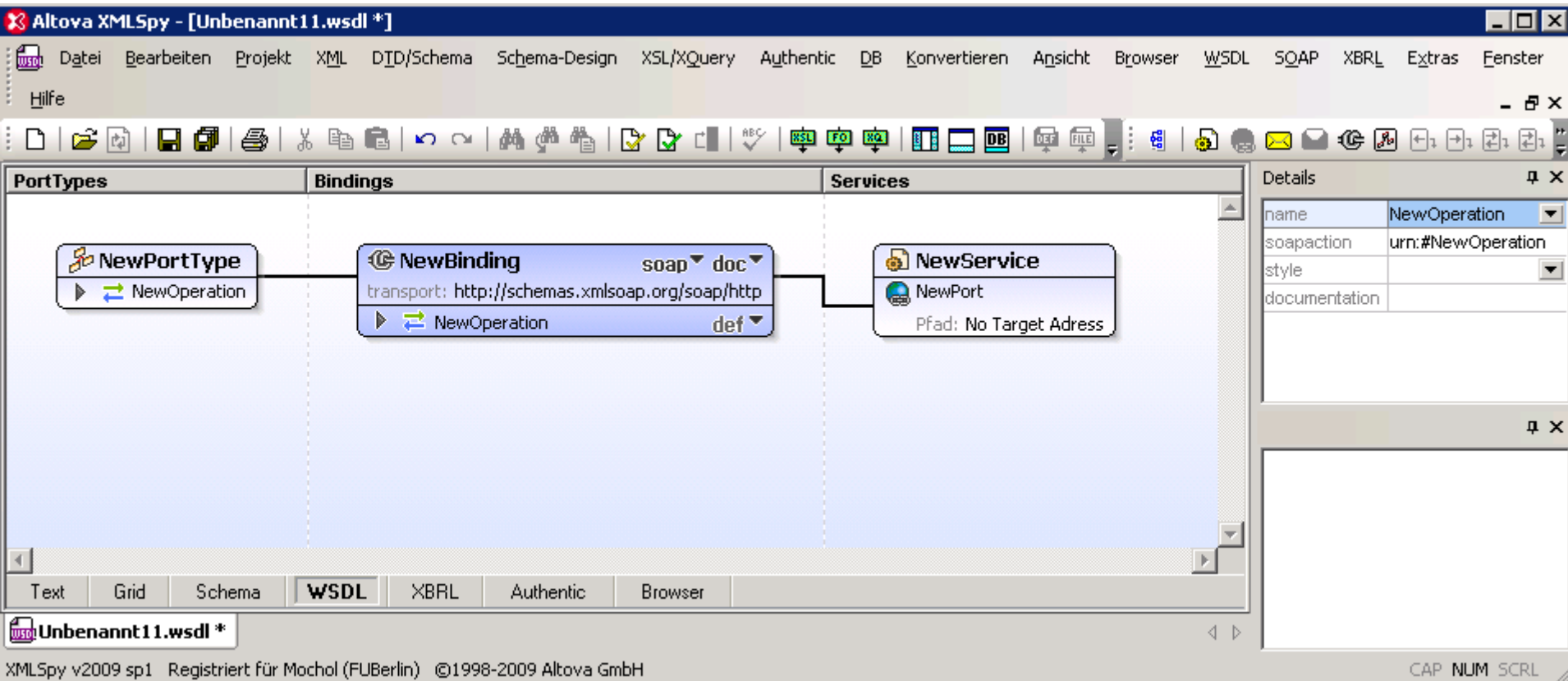
```

67      <Review>
68          <ASIN>3551566666</ASIN>
69          <Rating>3</Rating>
70          <HelpfulVotes>4</HelpfulVotes>
71          <CustomerId>AWYY13WN6P1K9</CustomerId>
72          <Reviewer>
73              <CustomerId>AWYY13WN6P1K9</CustomerId>
74              <Name>helmut seeger</Name>
75              <Nickname>liberaler</Nickname>
76              <Location>karlsruhe</Location>
77          </Reviewer>
78          <TotalVotes>8</TotalVotes>
79          <Date>2007-08-04</Date>
80          <Summary>überraschend positiver Gesamtklang</Summary>
81          <Content>Trotz des theatralischen Endes und Dumbledores Abgang am Ende des Bandes ist im
            "Halbblutprinzen" ein insgesamt überraschend positiver Gesamtklang und eine harmonische Atmosphäre gelungen, die
            man nach den düsteren Vorgängerbänden kaum erwartet hätte. Auch mit diesem Band hat die Autorin ein gelungenes
            Kinderbuch auch für Erwachsene vorgelegt und bastelt weiter an ihrem Mythos.
82          &lt;br /&gt;Das Buch bietet Unterhaltung auf mehreren Ebenen: erstmals widmet es sich über weite Strecken einem
            angemessenen pubertären Liebesgeplänkel - schließlich sind die Protagonisten im entsprechenden Alter. Allerdings
            immer getreulich dem Grundsatz "No Sex please, we're british". Knutscherei ist das Äußerste, zu dem die Beteiligten in
            der Lage scheinen.
83          &lt;br /&gt;Hauptsächlich jedoch schildert der Band die eigentliche und persönliche Geschichte Voldemorts und wie
            dieser seine Seele in "Horkruxen" überleben lassen konnte und bietet in diesem Zusammenhang eine Reihe
            Erläuterungen zu der gesamten Vorgeschichte der früheren Bände. Auch hier beweist Rowling wieder ausführliches
            Verständnis psychologischer Zusammenhänge - nebst Anspielungen auf psychedelische Erlebnisse unter Drogeneinfluss.
84          &lt;br /&gt;Die gesamte Geschichte ist wie immer flüssig, humorvoll und mit viel Einfühlungsvermögen in ihre jugendliche
            Protagonisten erzählt und auf jeden Fall pointierter als der etwas zerfahrene Band 5.
85          &lt;br /&gt;Als müsste Harry Potter für die Entbehrungen der vergangenen Bände entschädigt werden, wird er hier
            ausgiebig als Held gefeiert.
86          &lt;br /&gt;Insgesamt bietet der Band nicht mehr allzu viel an Neuem und Überraschungen (abgesehen vom sehr
            temporeichen, temperamentvollen und einfallsreichen letzten Drittel), zehrt aber sehr gekonnt vom Arsenal der ersten
            Bände.</Content>
87      </Review>
    
```

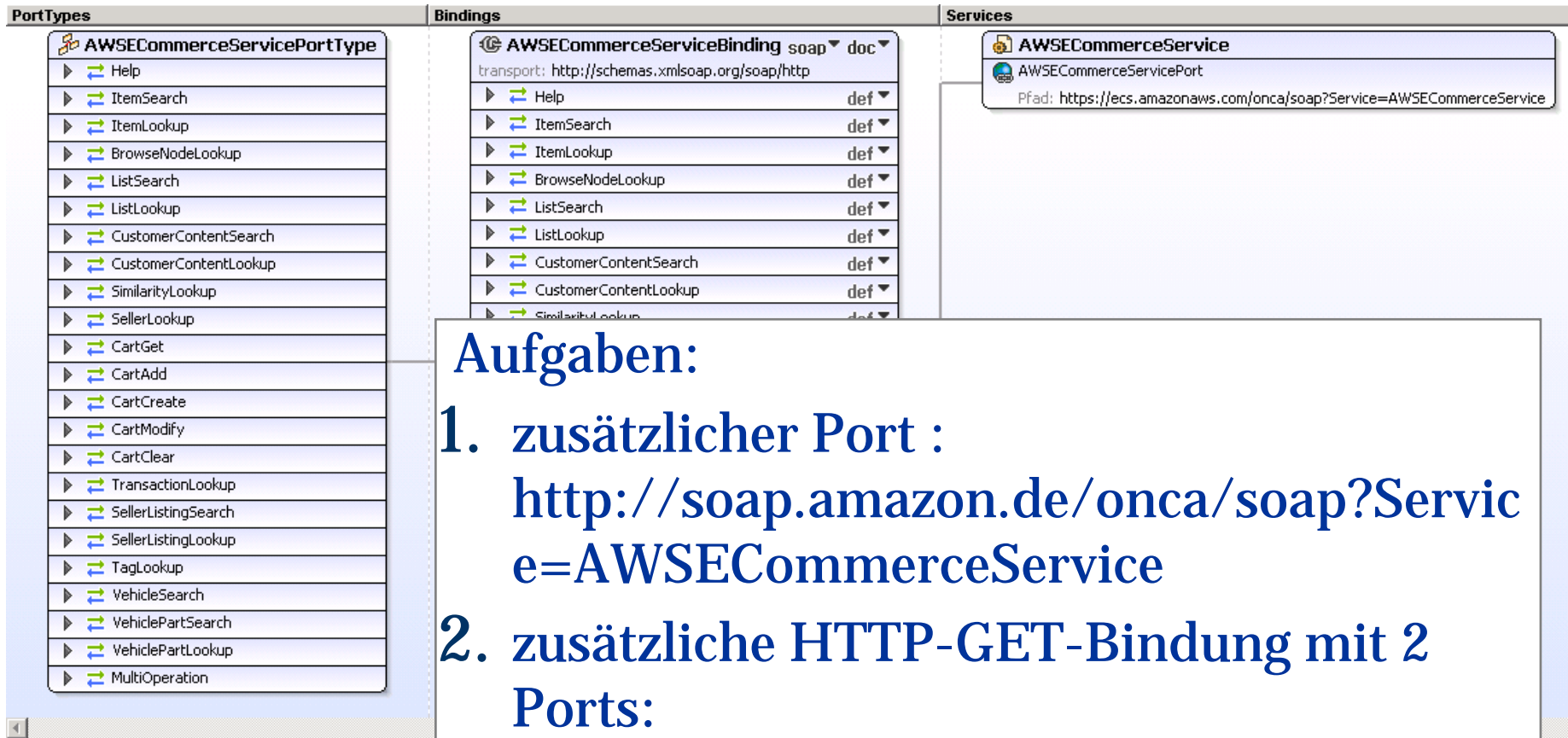


## Musterlösung des Übungsblattes 6

# Leere WSDL



The screenshot shows the Altova XMLSpy interface for a blank WSDL file named "Unbenannt11.wsdl". The main workspace is divided into three columns: PortTypes, Bindings, and Services. In the PortTypes column, there is a "NewPortType" component with a sub-component "NewOperation". In the Bindings column, there is a "NewBinding" component with a "transport" attribute set to "http://schemas.xmlsoap.org/soap/http" and a sub-component "NewOperation". In the Services column, there is a "NewService" component with a "NewPort" sub-component and a "Pfad: No Target Adress" attribute. The "Details" panel on the right shows the properties for the selected "NewOperation" component, including "name" (NewOperation), "soapaction" (urn:#NewOperation), "style", and "documentation". The bottom status bar indicates "XMLSpy v2009 sp1 Registriert für Mochol (FU Berlin) ©1998-2009 Altova GmbH".



The screenshot shows a WSDL editor with three main sections: PortTypes, Bindings, and Services. The PortTypes section lists various operations like Help, ItemSearch, ItemLookup, BrowseNodeLookup, ListSearch, ListLookup, CustomerContentSearch, CustomerContentLookup, SimilarityLookup, SellerLookup, CartGet, CartAdd, CartCreate, CartModify, CartClear, TransactionLookup, SellerListingSearch, SellerListingLookup, TagLookup, VehicleSearch, VehiclePartSearch, VehiclePartLookup, and MultiOperation. The Bindings section shows the AWSECommerceServiceBinding with a transport of http://schemas.xmlsoap.org/soap/http and lists the same operations as the PortTypes section. The Services section shows the AWSECommerceService with the port type AWSECommerceServicePort and the endpoint Pfad: https://ecs.amazonaws.com/onca/soap?Service=AWSECommerceService.

## Aufgaben:

1. zusätzlicher Port :  
<http://soap.amazon.de/onca/soap?Service=AWSECommerceService>
2. zusätzliche HTTP-GET-Bindung mit 2 Ports:  
liefert XML (und nicht SOAP) zurück

# 1) Zusätzlicher Port

PortTypes	Bindings	Services
<b>AWSECommerceServicePortType</b> <ul style="list-style-type: none"><li>Help</li><li>ItemSearch</li><li>ItemLookup</li><li>BrowseNodeLookup</li><li>ListSearch</li><li>ListLookup</li><li>CustomerContentSearch</li><li>CustomerContentLookup</li><li>SimilarityLookup</li><li>SellerLookup</li><li>CartGet</li><li>CartAdd</li><li>CartCreate</li><li>CartModify</li><li>CartClear</li><li>TransactionLookup</li><li>SellerListingSearch</li><li>SellerListingLookup</li><li>TagLookup</li><li>VehicleSearch</li><li>VehiclePartSearch</li><li>VehiclePartLookup</li><li>MultiOperation</li></ul>	<b>AWSECommerceServiceBinding soap doc</b> transport: http://schemas.xmlsoap.org/soap/http <ul style="list-style-type: none"><li>Help def</li><li>ItemSearch def</li><li>ItemLookup def</li><li>BrowseNodeLookup def</li><li>ListSearch def</li><li>ListLookup def</li><li>CustomerContentSearch def</li><li>CustomerContentLookup def</li><li>SimilarityLookup def</li><li>SellerLookup def</li><li>CartGet def</li><li>CartAdd def</li><li>CartCreate def</li><li>CartModify def</li><li>CartClear def</li><li>TransactionLookup def</li><li>SellerListingSearch def</li><li>SellerListingLookup def</li><li>TagLookup def</li><li>VehicleSearch def</li></ul>	<b>AWSECommerceService</b> <ul style="list-style-type: none"><li>AWSECommerceServicePort Pfad: https://ecs.amazonaws.com/onca/soap?Service=AWSECommerceService</li><li><b>AWSECommerceServicePortDE</b> Pfad: <b>http://soap.amazon.de/onca/soap?Service=AWSECommerceService</b></li></ul>

```
<service name="AWSECommerceService">  
  <port name="AWSECommerceServicePort" binding="tns:AWSECommerceServiceBinding">  
    <port name="AWSECommerceServicePortDE" binding="tns:AWSECommerceServiceBinding">  
      <soap:address location="http://soap.amazon.de/onca/soap?Service=AWSECommerceService"/>  
    </port>  
  </port>  
</service>
```

## 2) Zusätzliche HTTP-GET-Bindung (I)

AWSECommerceServicePortType	
▶	Help
▶	ItemSearch
▶	ItemLookup
▶	BrowseNodeLookup
▶	ListSearch
▶	ListLookup
▶	CustomerContentSearch
▶	CustomerContentLookup
▶	SimilarityLookup
▶	SellerLookup
▶	CartGet
▶	CartAdd
▶	CartCreate
▶	CartModify
▶	CartClear
▶	TransactionLookup
▶	SellerListingSearch
▶	SellerListingLookup
▶	TagLookup
▶	VehicleSearch
▶	VehiclePartSearch
▶	VehiclePartLookup
▶	MultiOperation

AWSECommerceServiceBinding soap doc	
transport: http://schemas.xmlsoap.org/soap/http	
▶	Help def
▶	ItemSearch def
▶	ItemLookup def
▶	BrowseNodeLookup def
▶	ListSearch def
▶	ListLookup def
▶	CustomerContentSearch def
▶	CustomerContentLookup def
▶	SimilarityLookup def
▶	SellerLookup def
▶	CartGet def
▶	CartAdd def
▶	CartCreate def
▶	CartModify def
▶	CartClear def
▶	TransactionLookup def
▶	SellerListingSearch def
▶	SellerListingLookup def
▶	TagLookup def
▶	VehicleSearch def
▶	VehiclePartSearch def
▶	VehiclePartLookup def
▶	MultiOperation def

AWSECommerceService	
▶	AWSECommerceServicePort
Pfad: https://ecs.amazonaws.com/onca/soap?Service=AWSECommerceService	
▶	AWSECommerceServicePortDE
Pfad: http://soap.amazon.de/onca/soap?Service=AWSECommerceService	

- PortType einfügen
- Binding einfügen**
- Service einfügen
- Message einfügen
- WSDL-Dokument neu parsen

## 2) Zusätzliche HTTP-GET-Bindung (II)

AWSECommerceServicePortType	
▶	Help
▶	ItemSearch
▶	ItemLookup
▶	BrowseNodeLookup
▶	ListSearch
▶	ListLookup
▶	CustomerContentSearch
▶	CustomerContentLookup
▶	SimilarityLookup
▶	SellerLookup
▶	CartGet
▶	CartAdd
▶	CartCreate
▶	CartModify
▶	CartClear
▶	TransactionLookup
▶	SellerListingSearch
▶	SellerListingLookup
▶	TagLookup
▶	VehicleSearch
▶	VehiclePartSearch
▶	VehiclePartLookup
▶	MultiOperation

AWSECommerceServiceBinding soap doc	
transport: http://schemas.xmlsoap.org/soap/http	
▶	Help def
▶	ItemSearch def
▶	ItemLookup def
▶	BrowseNodeLookup def
▶	ListSearch def
▶	ListLookup def
▶	CustomerContentSearch def
▶	CustomerContentLookup def
▶	SimilarityLookup def
▶	SellerLookup def
▶	CartGet def
▶	CartCreate def

AWSECommerceServiceBinding http <sup>get</sup>	
▶	Help
▶	ItemSearch
▶	ItemLookup
▶	BrowseNodeLookup
▶	ListSearch
▶	ListLookup
▶	CustomerContentSearch
▶	CustomerContentLookup
▶	SimilarityLookup
▶	SellerLookup
▶	CartGet
▶	CartAdd
▶	CartCreate

```
<binding name="AWSECommerceServiceBinding" type="tns:AWSECommerceServicePortType">  
  <http:binding verb="GET"/>  
  ...  
</binding>
```

- a) Was bedeutet das type-Attribut hier?
- b) Wozu der Bindung einen Namen geben?
- c) Reicht `<http:binding verb="GET"/>` aus?
- d) Was muss in der Bindung noch festgelegt werden?

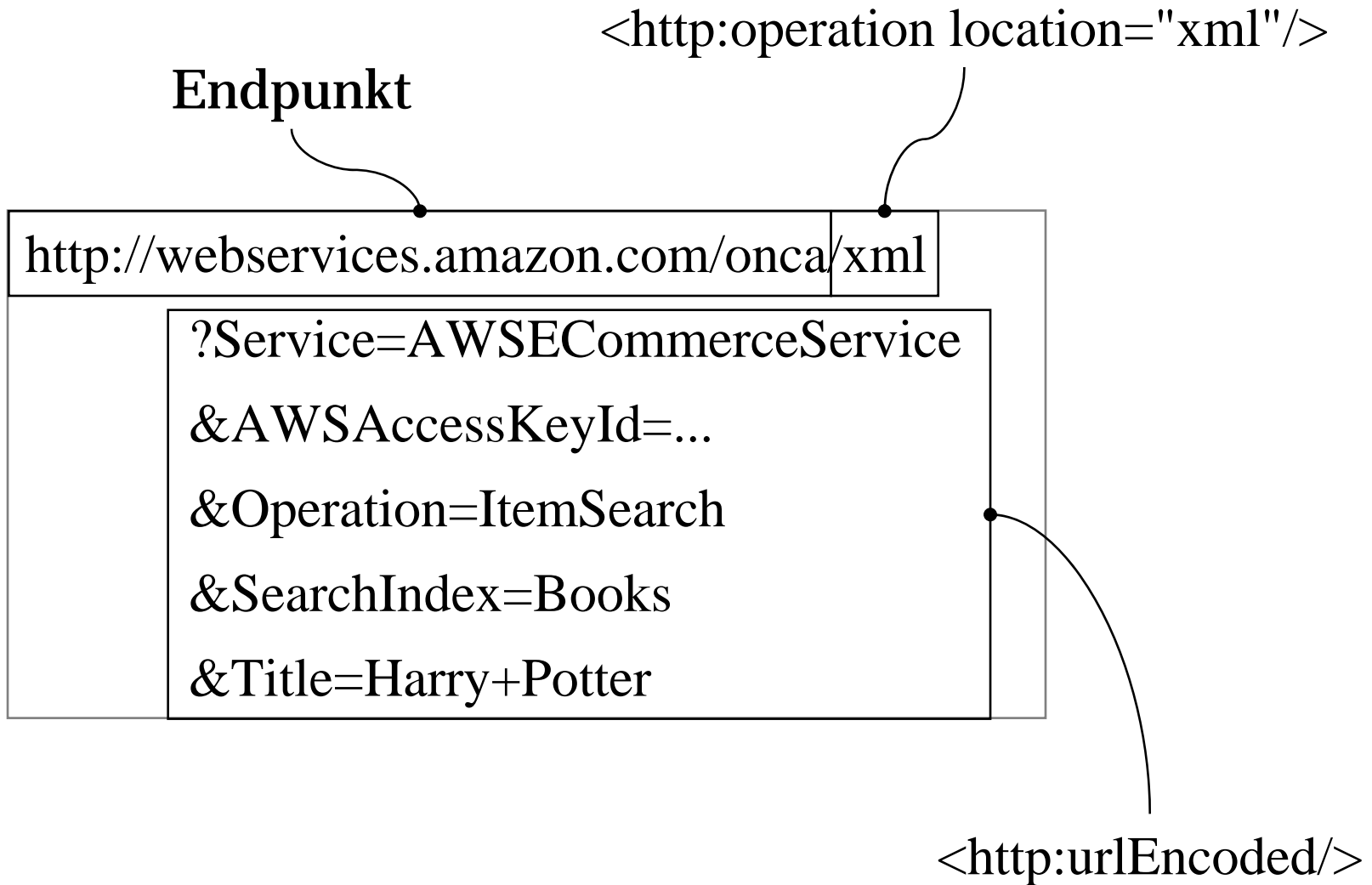
```
<binding name="AWSECommerceServiceRestBinding"
  type="tns:AWSECommerceServicePortType">
  <http:binding verb="GET"/>
  ...
  <operation name="ItemSearch">
    <http:operation location="xml"/>
    <input>
      <http:urlEncoded/>
    </input>
    <output>
      <mime:mimeXml/>
    </output>
  </operation>
  ...
</binding>
```

/xml wird an  
Endpunkt  
angehängt

?part1=value&...  
wird zusätzlich  
angehängt

URL = *Endpunkt/xml?part1=value&...*

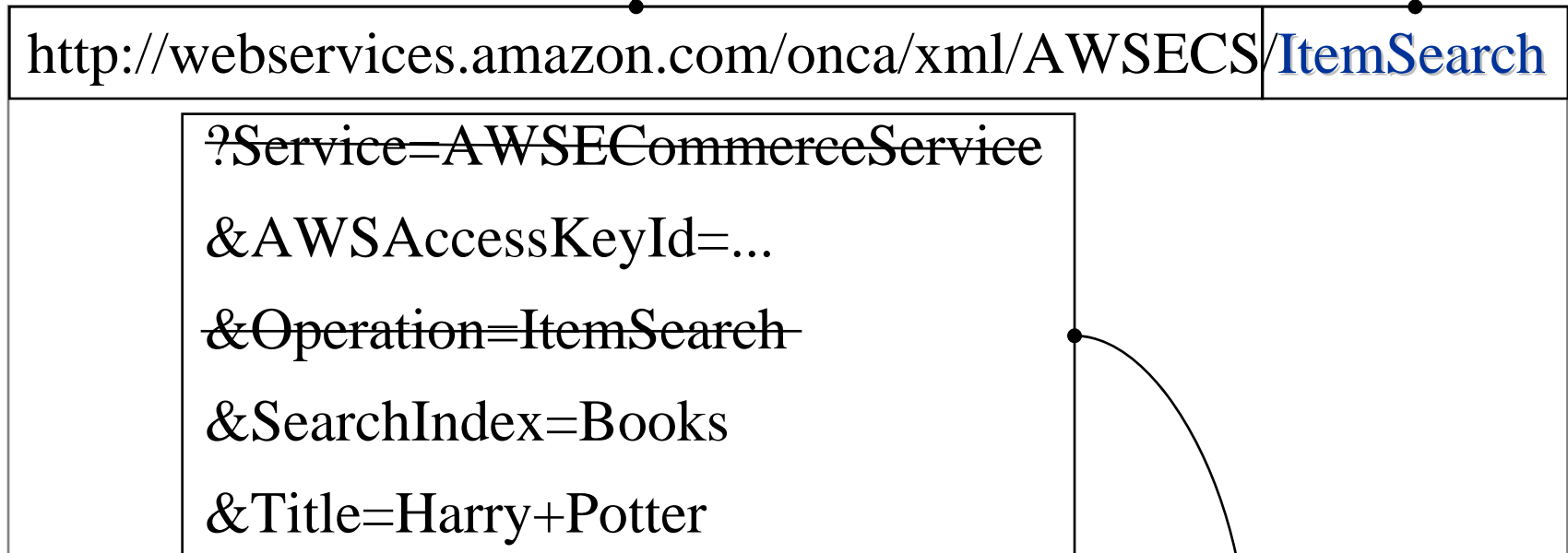
# Beispiel



# Mögliche Alternative (fiktiv)

<http:operation location="ItemSearch"/>

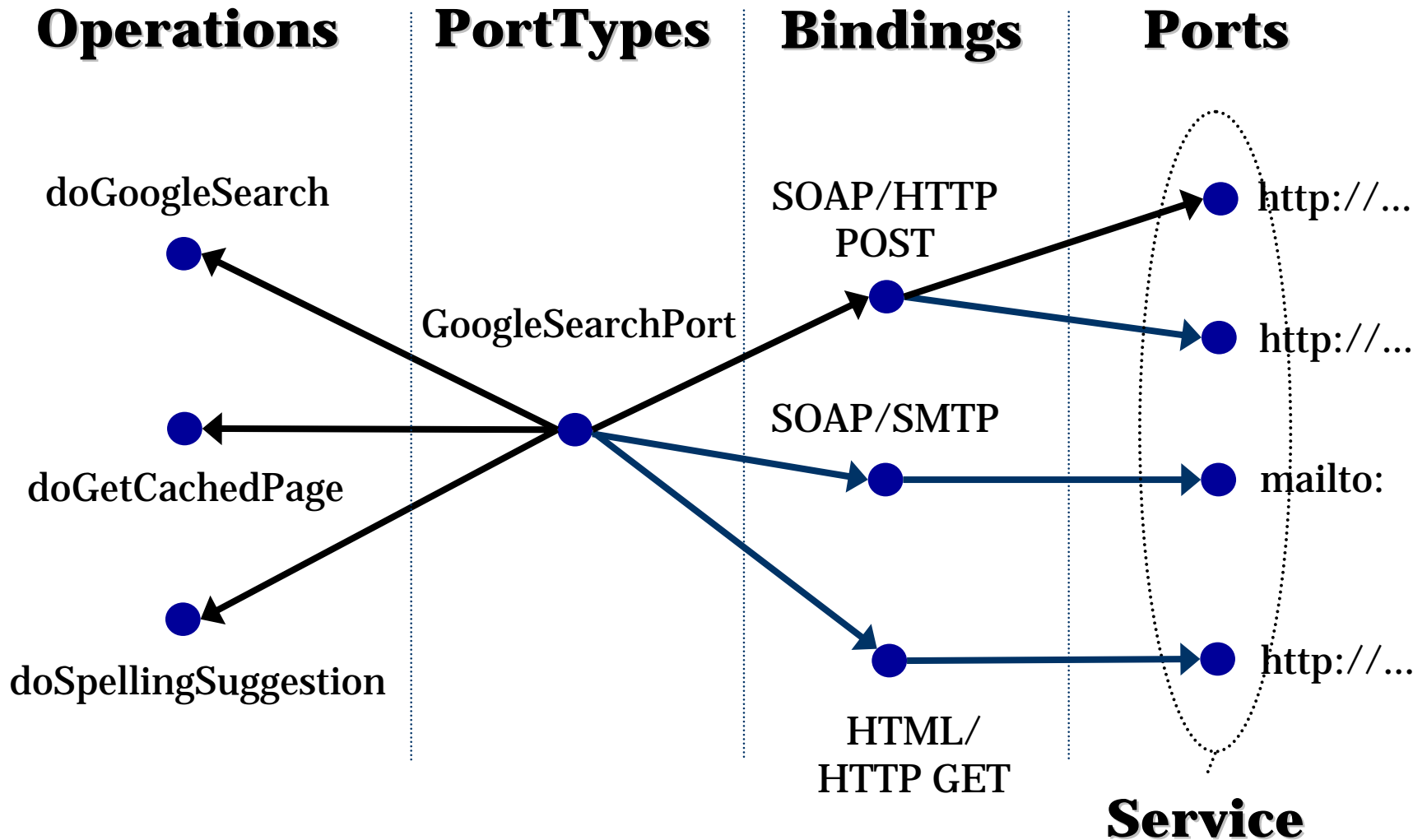
Endpunkt



<http:urlEncoded/>

```
<binding name="AWSECommerceServiceRestBinding"
  type="tns:AWSECommerceServicePortType">
  <http:binding verb="GET"/>
  ...
  <operation name="ItemSearch">
    <http:operation location="xml"/>
    <input>
      <http:urlEncoded/>
    </input>
    <output>
      <mime:mimeXml/>
    </output>
  </operation>
  ...
</binding>
```

Antwort ist  
XML, nicht  
SOAP



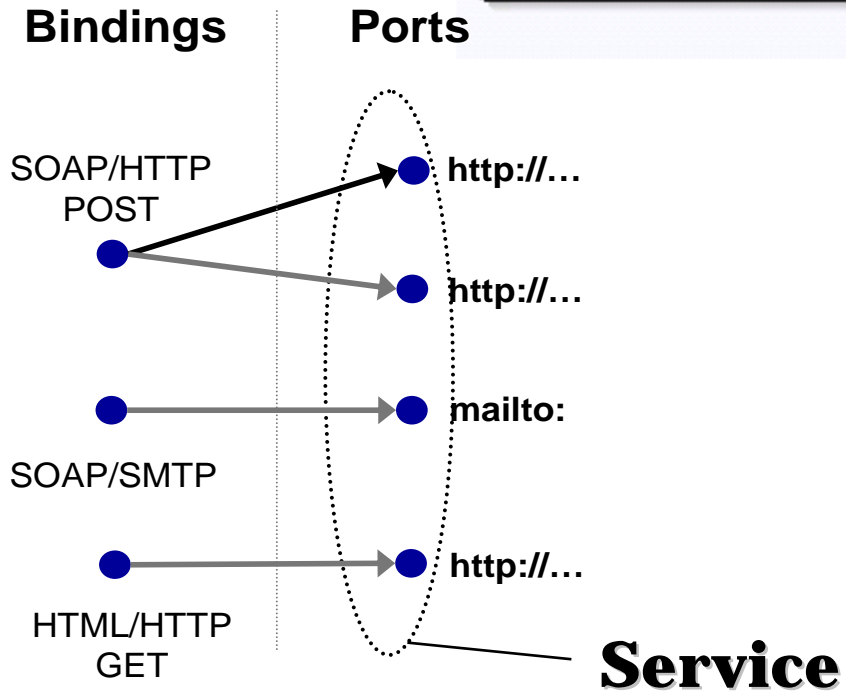
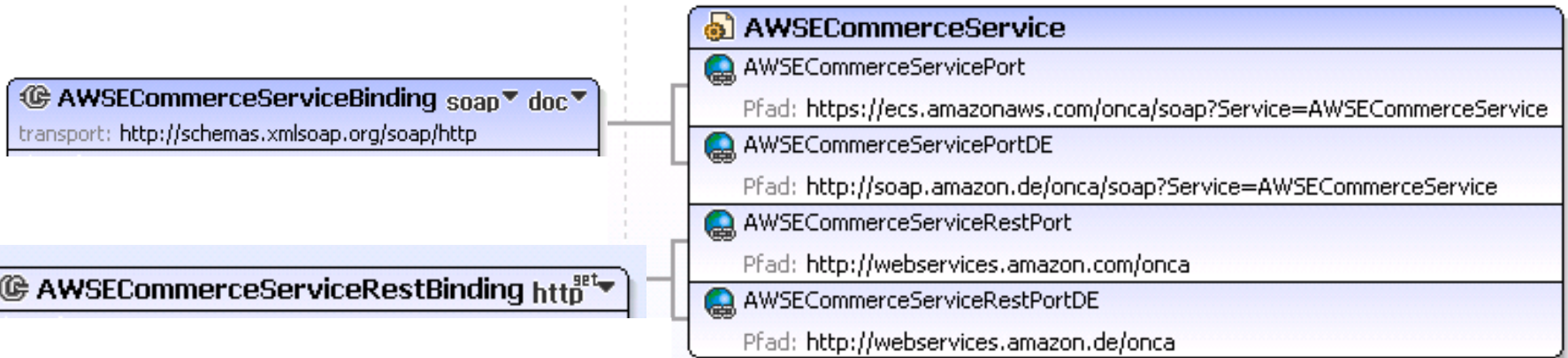
# Und die zugehörigen Ports

die „alten“ Ports

```
<service name="AWSECommerceService">  
  <port name="AWSECommerceServicePort" binding="tns:AWSECommerceServiceBinding">  
    <soap:address location="https://ecs.amazonaws.com/onca/soap?Service=AWSECommerceService"/>  
  </port>  
  <port name="AWSECommerceServicePortDE" binding="tns:AWSECommerceServiceBinding">  
    <soap:address location="http://soap.amazon.de/onca/soap?Service=AWSECommerceService"/>  
  </port>  
  <port name="AWSECommerceServiceRestPort" binding="tns:AWSECommerceServiceBinding">  
    <http:address location="http://webservices.amazon.com/onca"/>  
  </port>  
  <port name="AWSECommerceServiceRestPortDE" binding="tns:AWSECommerceServiceBinding">  
    <http:address location="http://webservices.amazon.de/onca"/>  
  </port>  
</service>
```

die neu eingefügten Ports

# Und die zugehörigen Ports (graphisch)



# <http:urlEncoded/> - klappt's?

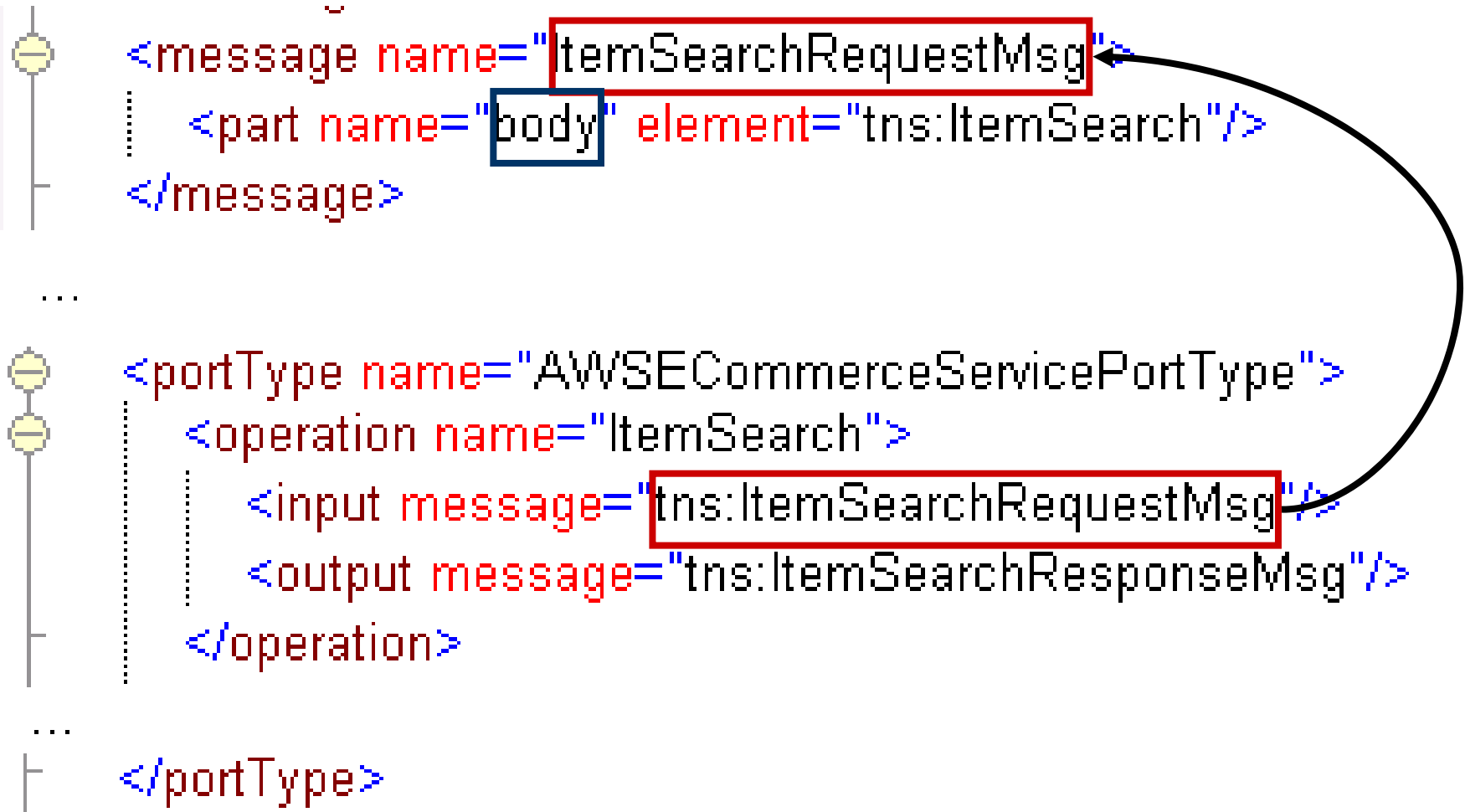
```

<binding name="AWSECommerceServiceRestBinding"
  type="tns:AWSECommerceServicePortType">
  <http:binding verb="GET"/>
  ...
  <operation name="ItemSearch">
    <http:operation location="xml"/>
    <input>
      <http:urlEncoded/>
    </input>
    <output>
      <mime:mimeXml/>
    </output>
  </operation>
  ...
</binding>
  
```

*?part1=value&...*  
wird zusätzlich  
angehängt

*part1, partn:*  
jeweils part-Name der abstrakten  
Eingangsnachricht (message) von  
ItemSearch

# part = body!



# Datentyp / Nachricht / Porttyp

```
<types>
  <xsd:schema xmlns:xsd="..." xmlns:tns="..." targetNamespace="...">
    <xsd:complexType name="GoogleSearchResult">
      ...
    </xsd:complexType>
  </schema>
</types>
```

Definition des Datentyps

```
<message name="doGoogleSearch">...</message>
<message name="doGoogleSearchResponse">
  <part name="return" type="tns:GoogleSearchResult"/>
</message>
```

Definition einer abstrakten Nachricht

```
<portType>
  <operation name="doGoogleSearch">
    <input message="tns:doGoogleSearch"/>
    <output message="tns:doGoogleSearchResponse"/>
  </operation>
  ...
</portType>
```

Definition einer abstrakten Schnittstelle

portType  
message  
types

- part-Name der Eingangsnachricht von ItemSearch ist **body**
- ⇒ URL = `http://webservices.amazon.com/onca/xml?body=value`
- sollte aber z.B. so aussehen:

```
http://webservices.amazon.com/onca/xml
    ?Service=AWSECommerceService
    &AWSAccessKeyId=...
    &Operation=ItemSearch
    &SearchIndex=Books
    &Title=Harry+Potter
```

# Und gleich noch ein Problem!

```
<message name="ItemSearchRequestMsg">  
  <part name="body" element="tns:ItemSearch"/>  
</message>
```

- <http://webservices.amazon.com/onca/xml?body=value>
- *value* ist Element ItemSearch

```
<xs:element name="ItemSearch">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="MarketplaceDomain" type="xs:string" minOccurs="0"/>
      <xs:element name="AWSAccessKeyId" type="xs:string" minOccurs="0"/>
      <xs:element name="SubscriptionId" type="xs:string" minOccurs="0"/>
      ...
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

- <http://webservices.amazon.com/onca/xml?body=value>
- Wieso ist obige Definition von ItemSearch ein Problem?

# Die Ursache des Problems!

Ein part für  
alle  
Parameter

```
<message name="ItemSearchRequestMsg">  
  <part name="body" element="tns:ItemSearch"/>  
</message>
```

```
<xs:element name="ItemSearch">  
  <xs:complexType>  
    <xs:sequence>  
      <xs:element name="MarketplaceDomain" type="xs:string" minOccurs="0"/>  
      <xs:element name="AWSAccessKeyId" type="xs:string" minOccurs="0"/>  
      <xs:element name="SubscriptionId" type="xs:string" minOccurs="0"/>  
      ...  
    </xs:sequence>  
  </xs:complexType>  
</xs:element>
```

zwei unterschiedliche Modellierungen

1. mehrere **parts**:

```
<message name="doGoogleSearchResponse">  
  <part name="param1" element="tns:param1"/>  
  <part name="param2" element="tns:param2"/>  
</message>
```

2. ein **part** mit komplexen Datentyp:

```
<message name="doGoogleSearchResponse">  
  <part name="return" type="tns:complexType"/>  
</message>
```

tns:complexType könnte z.B. 2 Parameter enthalten

# Zweiter Versuch!

<http://webservices.amazon.com/onca/xml>

Ziel

```
?Service=AWSECommerceService
&AWSAccessKeyId=...
&Operation=ItemSearch
&SearchIndex=Books
&Title=Harry+Potter
```

```
<message name="ItemSearchRequestMsg">
  <part name="body" element="tns:ItemSearch"/>
</message>
```

```
<message name="ItemSearchRequestMsg">
  <part name="Service" type="tns:AWSECS-String"/>
  <part name="Operation" type="tns:ItemSearch-String"/>
  <part name="MarketplaceDomain" type="xs:string"/>
  <part name="AWSAccessKeyId" type="xs:string"/>
  <part name="SubscriptionId" type="xs:string" />
  ...
</message>
```

neu!

vorher  
Kind-  
Elemente

# Das Ergebnis: Sieht doch gut aus!

```
http://webservices.amazon.com/onca/xml/  
?Service=AWSECommerceService  
&Operation=ItemSearch  
&MarketplaceDomain=string  
&AWSAccessKeyId=string  
&SubscriptionId=string  
...
```

# Und schon wieder ein Problem!

```
<message name="ItemSearchRequestMsg">  
  <part name="Service" type="tns:AWSECS-String"/>  
  <part name="Operation" type="tns:ItemSearch-String"/>  
  <part name="MarketplaceDomain" type="xs:string"/>  
  <part name="AWSAccessKeyId" type="xs:string"/>  
  <part name="SubscriptionId" type="xs:string" />
```

```
http://webservices.amazon.com/onca/xml/  
?Service=AWSECommerceService  
&Operation=ItemSearch  
&MarketplaceDomain=string  
&AWSAccessKeyId=string  
&SubscriptionId=string  
...
```

dieser Parameter  
z.B. sollte  
optional sein!

# Wie ein part optional machen?

```
<message name="ItemSearchRequestMsg">
  <part name="Service" type="tns:AWSECS-String"/>
  <part name="Operation" type="tns:ItemSearch-String"/>
  <part name="MarketplaceDomain" type="xs:string"/>
  <part name="AWSAccessKeyId" type="xs:string"/>
  <part name="SubscriptionId" type="xs:string" />
  ...
</message>
```

- **Antwort: Geht gar nicht!**
- ⇒ **wahrscheinlich Grund, warum diese Bindung in der ECS-WSDL nicht beschrieben wird!**



## Web Service Komposition

Web Service Komposition – Vereinigung bestehender Web Services und anderer Komponenten, um neue Prozesse zu erzeugen

- Bereitstellung von Mehrwertdiensten durch Zusammenschluss mehrerer Web Services
- Serviceorientierte Entwicklung von Systemen
- Ausnutzung wachsender Anzahl bestehender Online-Dienste
- plattformunabhängige Kommunikation zwischen Web Services
- basiert NICHT auf der physikalischen Integration von Komponenten

1. **proaktive** vs. **reaktive** Komposition
2. Konversation (conversation) & Konversationsunterstützung
3. (a) **Orchestration** vs. (b) **Choreography**  
(Orchestration/Choreography)

# 1. proaktive vs. reaktive Komposition

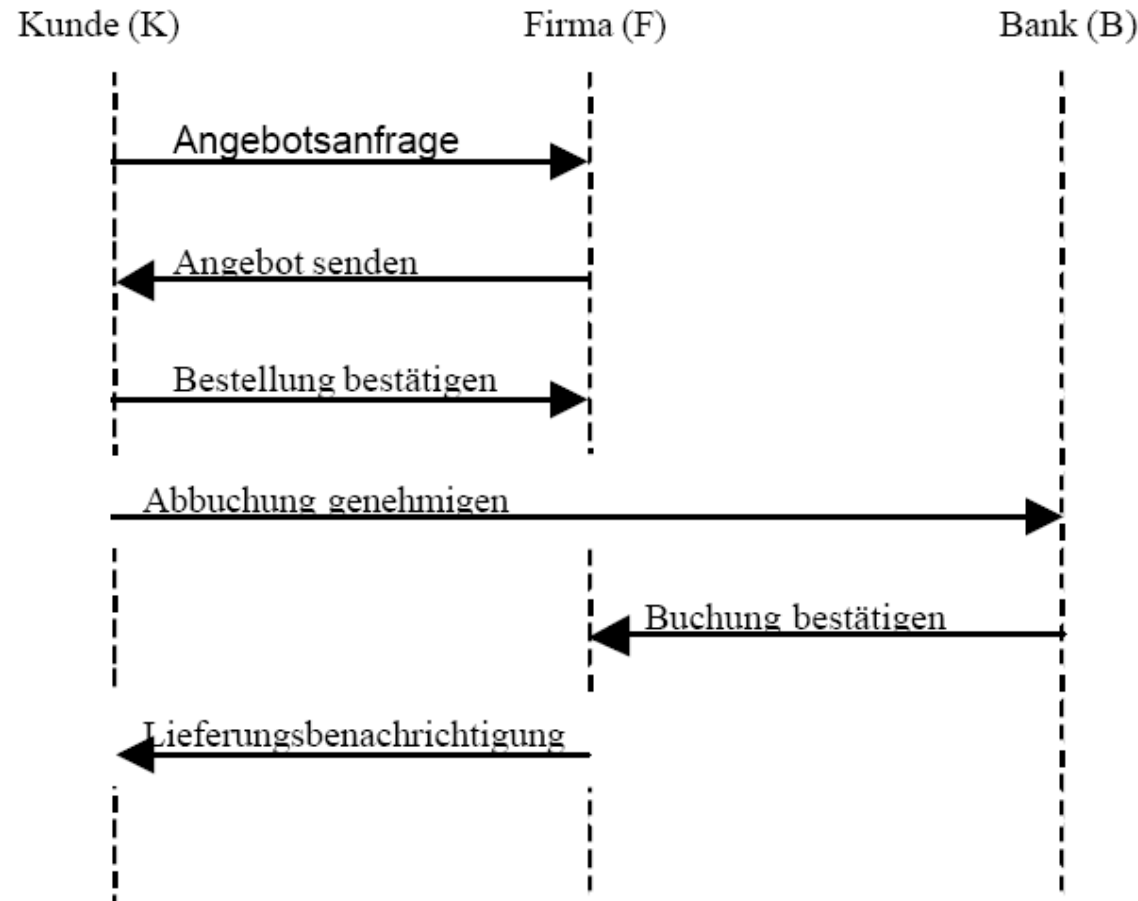
## proaktive Komposition

- alle verwendeten Services zur Entwicklungszeit bekannt
- geeignet, wenn die Anwendung
  - oft benutzt wird & stabil sein muss
  - lange Laufzeiten benötigt werden
  - die Geschwindigkeit eine Rolle spielt

## reaktive Komposition

- dynamisches Komponieren des neuen Dienstes
- erlaubt während der Laufzeit Optimierungen vorzunehmen
- geeignet, wenn der Dienst
  - nur selten nachgefragt wird
  - die einzelnen Komponenten nicht im Vorfeld bekannt sind

## 2. Nachrichtenabfolge (Konversation)



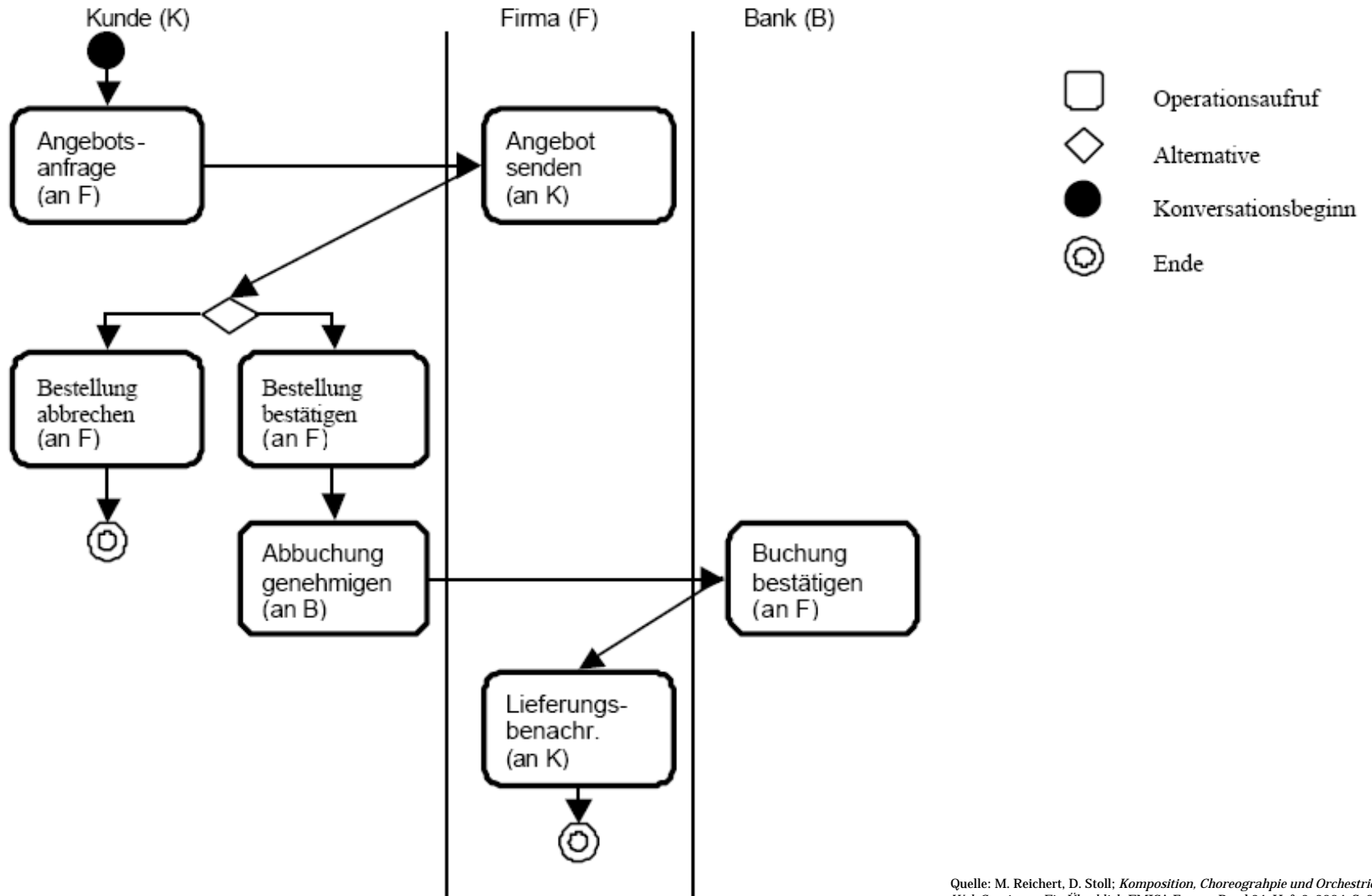
- **Abfolge von Operationen** zwischen Web Services
- **konkrete Kommunikation** von zwei oder mehreren Web Services
- **Kontext** erforderlich

Quelle: M. Reichert, D. Stoll; *Komposition, Choreographie und Orchestrierung von Web Services – Ein Überblick*, EMISA Forum, Band 24, Heft 2, 2004, S. 21-32

## 3a. Komposition: Choreography

- beschreibt aus **öffentlicher Sicht** die **zulässige Abfolge von Nachrichten** zwischen einem oder mehreren Partnern
- keine Angaben über die interne Prozesslogik der Teilnehmer
- kein zentraler Koordinator
- nicht als Prozess ausführbar
- jeder Web Service "weiss", mit wem er interagieren soll
- **Konversations-** oder **Koordinationsprotokoll** (*coordination protocol*) – Menge zulässiger Nachrichtenabfolgen

# Choreography – Beispiel

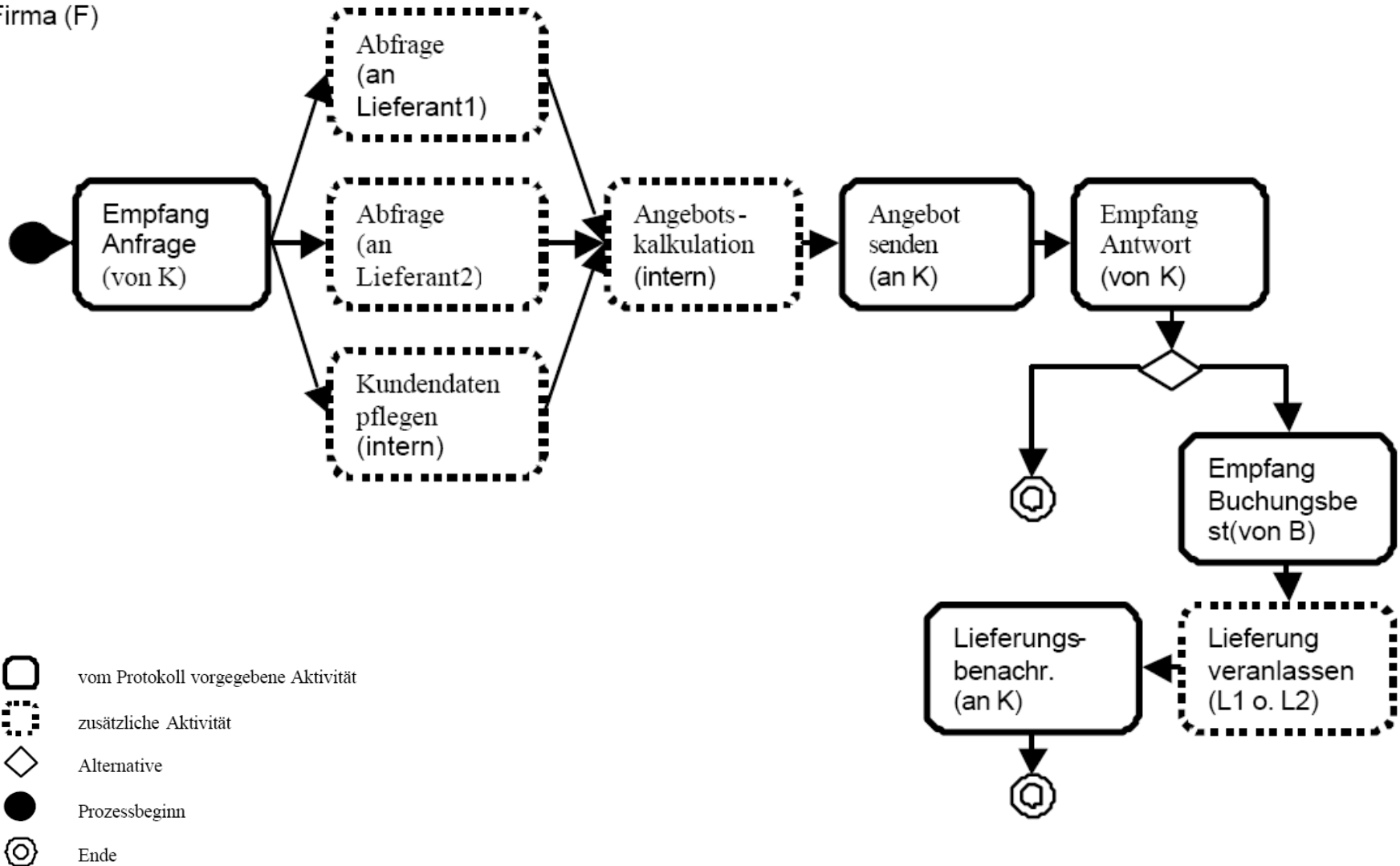


Quelle: M. Reichert, D. Stoll; *Komposition, Choreographie und Orchestrierung von Web Services – Ein Überblick*, EMISA Forum, Band 24, Heft 2, 2004, S. 21-32

- Kopplung einzelner Web Services, um einen komplexen Geschäftsprozess aufzubauen
- beschreibt die **Geschäftsprozesslogik aus der Sicht eines Teilnehmers**
- ein zentraler Prozess koordiniert die Operationen der Web Services, die in den Prozess eingebunden sind

# Orchestrierung – Beispiel

Firma (F)



Quelle: M. Reichert, D. Stoll; *Komposition, Choreographie und Orchestrierung von Web Services – Ein Überblick*, EMISA Forum, Band 24, Heft 2, 2004, S. 21-32

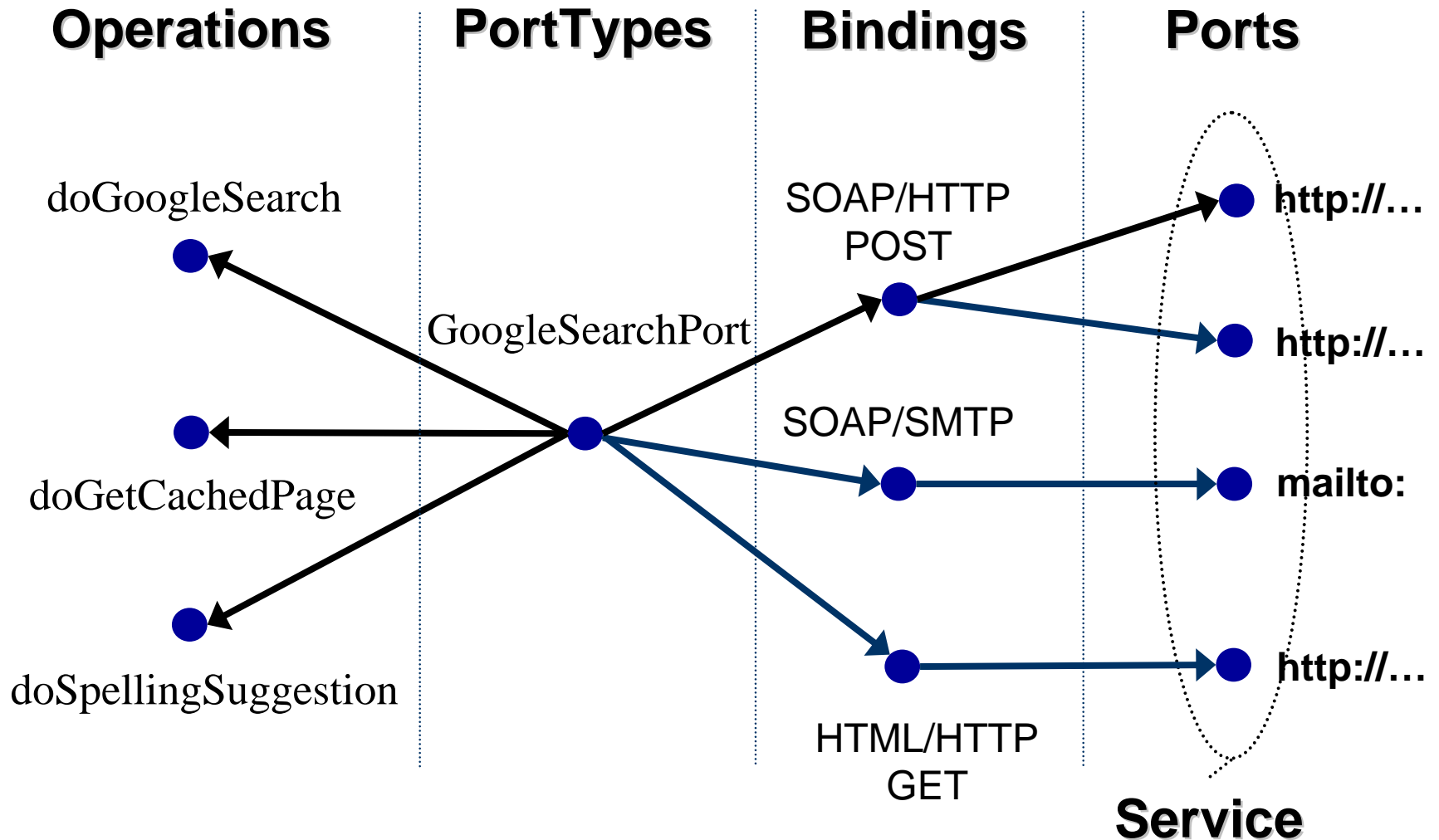


## Musterfragen

# 1. Musterfrage

## WSDL relates

1. Services to operations to port types to bindings.
2. Services to operations to bindings to port types.
3. Services to bindings to operations to port types.
4. Services to bindings to port types to operations.
5. Services to port types to operations to bindings.



## 2. Musterfrage

The four types of WSDL transmission primitives are:

1. Publish-subscribe, solicit-response, notify, update.
2. Notify, publish-subscribe, one-way, solicit.
3. Send-receive, send, receive, receive-send.
4. Solicit, notify, one-way, request-response.
5. Notification, request-response, solicit-response, one-way.

## Einweg (oneway)

```
<operation name="...">  
  <input message="..."/>  
</operation>
```

## Benachrichtigung (notification)

```
<operation name="...">  
  <output message="..."/>  
</operation>
```

## Anfrage-Antwort (request-response)

```
<operation name="...">  
  <input message="..."/>  
  <output message="..."/>  
</operation>
```

## Benachrichtigung-Antwort (solicit-response)

```
<operation name="...">  
  <output message="..."/>  
  <input message="..."/>  
</operation>
```



**Klausur → Organisatorisches**

## Organisatorisches

- 15 Juli um 14:00 **pünktlich**
- in Hörsaal Informatik

## Was sind die Voraussetzungen?

- Sie haben die **Folien der Vorlesung verstanden** und können dieses **Wissen anwenden**.
- Sie haben die **Übungen gemacht** und die Musterlösungen verstanden.
- Sie beherrschen den **Vorlesungsinhalt passiv und soweit aktiv**, dass Sie kleine Änderungen vornehmen können

## Struktur: Sie haben 90 Minuten um 90 Punkte zu erreichen!

- 20 - 24 Multiple Choice Fragen
  - **1 Punkt** für eine **vollständig** richtige Antwort (20-24 Punkte insgesamt)
  - mehrere richtige Antworten möglich
- 5-7 ausführliche Fragen
  - je Frage **8-10 Punkte** (66-70 Punkte insgesamt)

## Beispiel: Multiple Choice

- Ähnlich zu den Fragen, die in der Übung vorgestellt wurden
- Aufgepasst: nicht immer nur **eine** korrekte Antwort!

## Beispiel: ausführliche Fragen

- Typischerweise mit gegebenem XML Code
- Aufgabe: Korrektur, Änderung, Erweiterung des Codes.
- Oder: XML Instanz von Schema, XSLT Ausgabe usw.

<b>Punkte</b>	<b>Note</b>
<b><math>\geq 85,5</math></b>	<b>1,0</b>
<b><math>\geq 81</math></b>	<b>1,3</b>
<b><math>\geq 76,5</math></b>	<b>1,7</b>
<b><math>\geq 72</math></b>	<b>2,0</b>
<b><math>\geq 67,5</math></b>	<b>2,3</b>
<b><math>\geq 63</math></b>	<b>2,7</b>
<b><math>\geq 58,5</math></b>	<b>3,0</b>
<b><math>\geq 54</math></b>	<b>3,3</b>
<b><math>\geq 49,5</math></b>	<b>3,7</b>
<b><math>\geq 41</math></b>	<b>4,0</b>

- Fragen sind **nur auf Deutsch**
- **Antworten auf Deutsch oder Englisch** möglich
- Keine eigenen (Wörter-)Bücher, Papiere usw. erlaubt
- **Handy aus!**
- Während der Prüfung, dürfen Sie Hand hochheben wenn Sie Hilfe brauchen.
- Sitzordnung: jede zweite Reihe, jeder zweite Platz!
- Studentenausweis und auch Ausweis mit Foto mitbringen!

Weitere Fragen?



Das war's!

**Good Luck!**