

Vorlesung Wintersemester 2010/11

Konzepte und Anwendung von Workflowsystemen

Kapitel 8: Workflow Ausführungssprache BPEL

Lehrstuhl für Systeme der Informationsverwaltung, Prof. Böhm
Institut für Programmstrukturen und Datenorganisation (IPD)

Überblick Kapitel 8

Web Services Business Process Execution Language

- ◆ Historie
- ◆ Überblick
- ◆ Beispiel BPEL
- ◆ Sprachkonstrukte in BPEL
- ◆ Umsetzung von BPEL
- ◆ Mapping von BPMN -> BPEL

- ◆ Intalio Demo

Historie

- ◆ Ergebnis der Zusammenführung zweier Workflow-Sprachen:
 - XLANG (Microsoft); block-strukturiert;
 - WSFL (Web Services Flow Language, IBM); graph-basiert

- ◆ WS-BPEL (Web Services Business Process Execution Language), kurz auch BPEL

- ◆ BPEL4WS 1.0 August 2002 (Microsoft, IBM, BEA)
- ◆ WS-BPEL 1.1 Mai 2003 (Microsoft, IBM, BEA, SAP, Siebel); danach an OASIS übertragen
- ◆ WS-BPEL 2.0 Januar 2007 OASIS Standard

Überblick WS-BPEL

WS-BPEL

Historie

Überblick

Beispiel

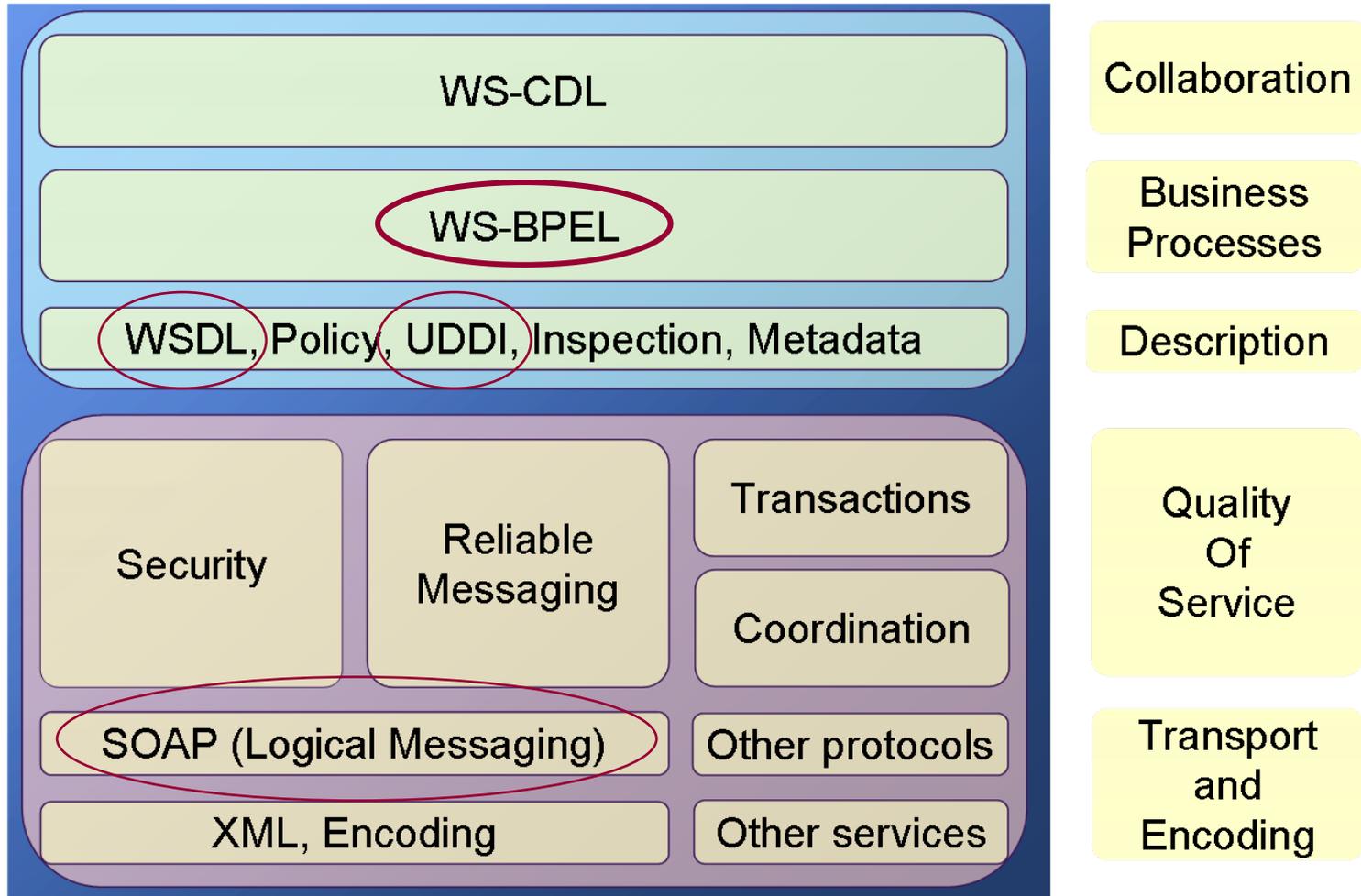
Sprachkonstrukte

Umsetzung

Mapping

- ◆ XML-basierte **Ausführungs**-Sprache zur Spezifikation von Geschäftsprozessen und Interaktionsprotokollen basierend auf Web Services
- ◆ Basiert auf verschiedene XML Standards (layer):
 - XML Schema 1.0
 - XPath 1.0
 - WSDL 1.1
- ◆ Einbindung und Koordination der „eigentlichen“ Aufgaben (-> Terminologie Modellierung: Aktivitäten) als Web Services (über WSDL)
- ◆ Ausführung eines BPEL-Prozesses durch Prozess-Engine

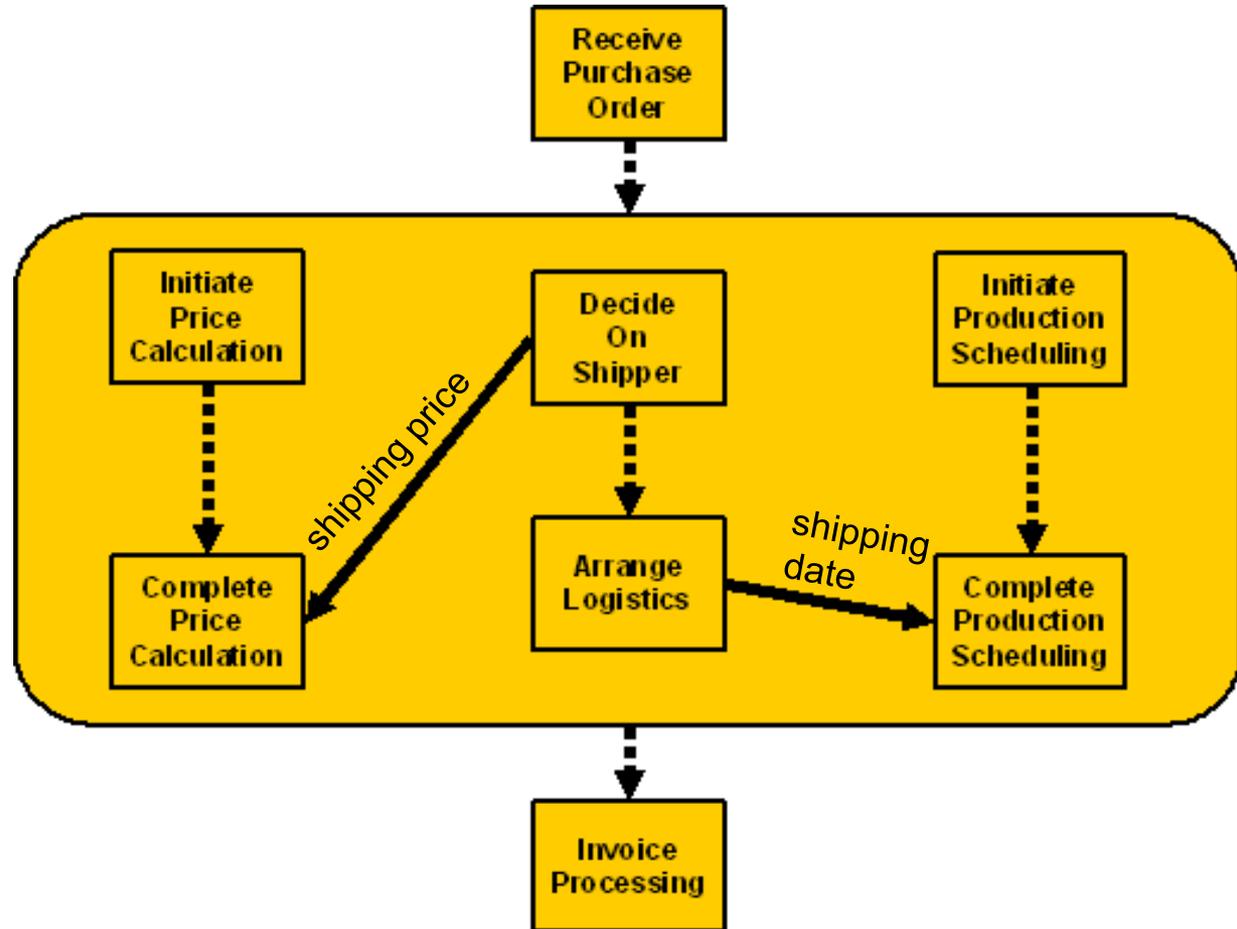
WS-BPEL im Web Service Stack (Wdh)



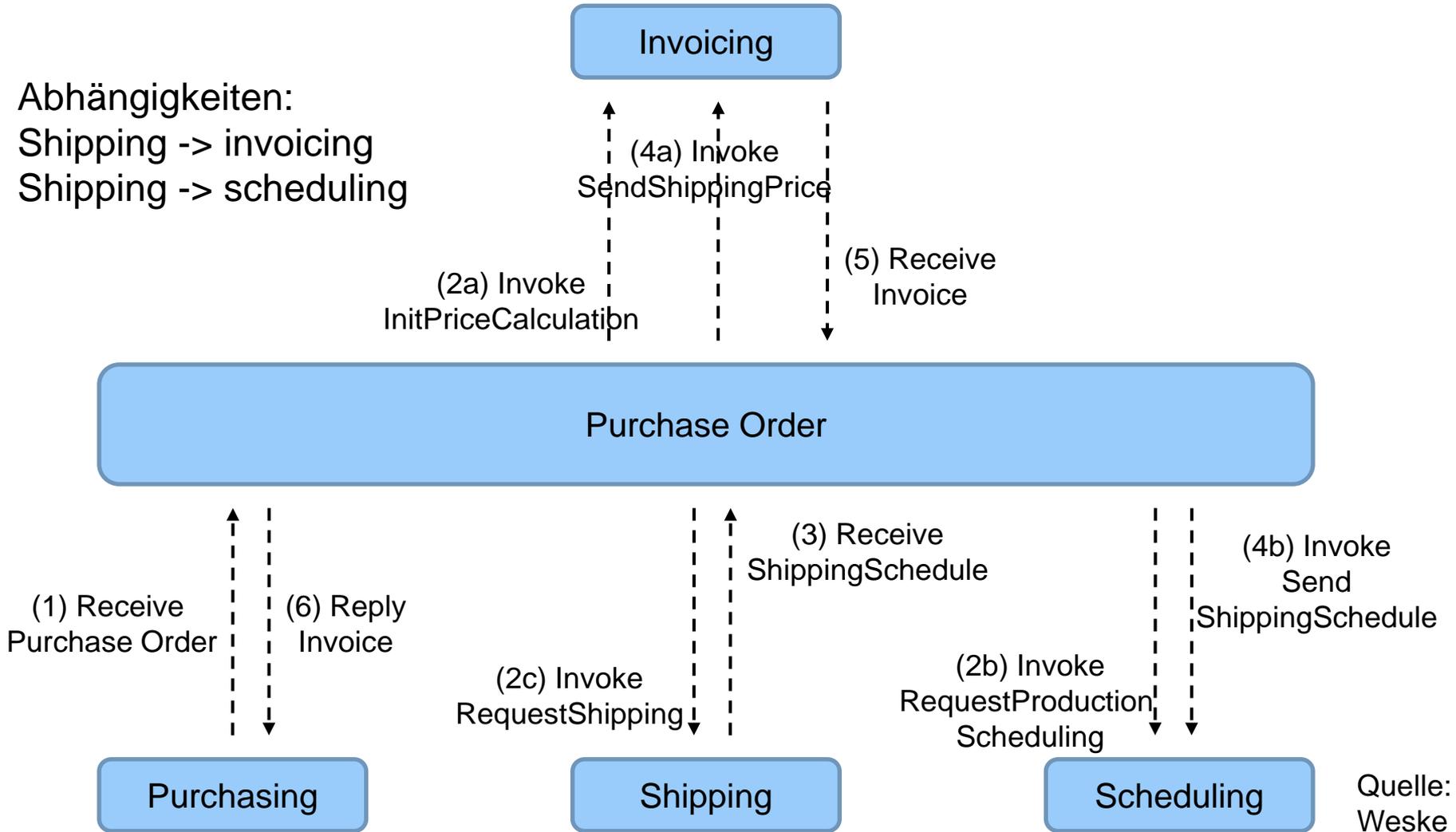
Einführungsbeispiel Auftragsabwicklung

WS-BPEL
Historie
Überblick
Beispiel
Sprachkonstrukte
Umsetzung
Mapping

Legende:
---> Sequenzfluss
-> Kontrollfluss

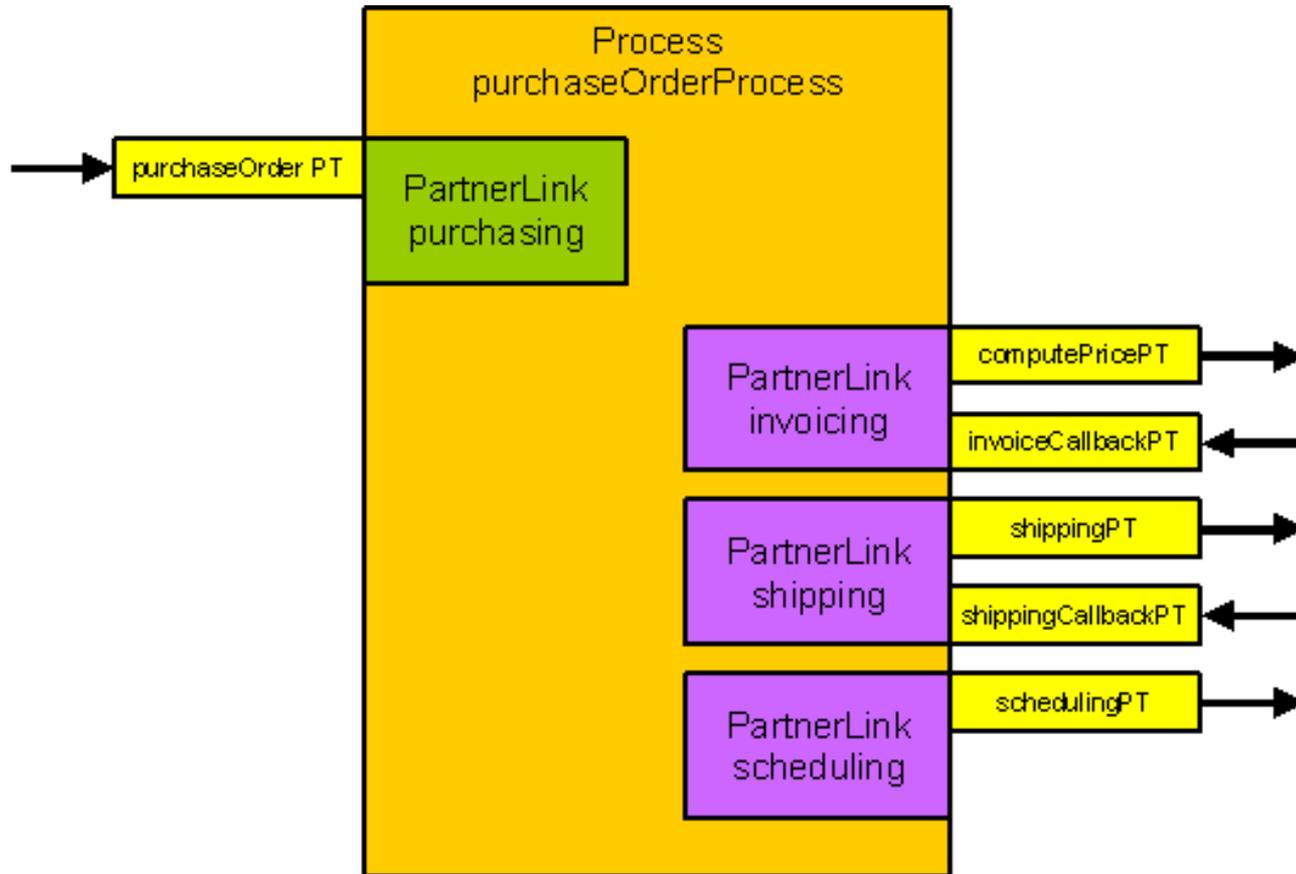


Aufrufe in BPEL am Beispiel Auftragsabwicklung



Quelle:
Weske

Interaktion und Choreographie am Beispiel



WSDL-Definition von „Port Types“ am Beispiel

```
<wsdl:definitions .....>
.....
<!-- portTypes supported by the purchase order process -->
  <wsdl:portType name="purchaseOrderPT">
    <wsdl:operation name="sendPurchaseOrder">
      <wsdl:input message="pos:POMessage" />
      <wsdl:output message="pos:InvMessage" />
      <wsdl:fault name="cannotCompleteOrder"
        message="pos:orderFaultType" />
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:portType name="invoiceCallbackPT">
    <wsdl:operation name="sendInvoice">
      <wsdl:input message="pos:InvMessage" />
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:portType name="shippingCallbackPT">
    <wsdl:operation name="sendSchedule">
      <wsdl:input message="pos:scheduleMessage" />
    </wsdl:operation>
```

WSDL Definition von Interfaces am Beispiel

```
<plnk:partnerLinkType name="purchasingLT">  
  <plnk:role name="purchaseService"  
    portType="pos:purchaseOrderPT" />  
</plnk:partnerLinkType>
```

```
<plnk:partnerLinkType name="invoicingLT">  
  <plnk:role name="invoiceService"  
    portType="pos:computePricePT" />  
  <plnk:role name="invoiceRequester"  
    portType="pos:invoiceCallbackPT" />  
</plnk:partnerLinkType>
```

```
<plnk:partnerLinkType name="shippingLT">  
  <plnk:role name="shippingService"  
    portType="pos:shippingPT" />  
  <plnk:role name="shippingRequester"  
    portType="pos:shippingCallbackPT" />  
</plnk:partnerLinkType>
```

<partnerLinkType>

- ◆ Darstellung von Interaktionen
- ◆ Bis zu zwei Rollen möglich
- ◆ Spezifikation der „Port Types“ der Rollen

(Vereinf.) BPEL-Spezifikation am Beispiel (1. Teil)

```
<process name="purchaseOrderProcess"
  targetNamespace="http://example.com/ws-bp/purchase"
  xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
  xmlns:ins="http://manufacturing.org/wsd/purchase">
  <documentation xml:lang="EN"> A simple example of WS-BPEL</documentation>
```

Ausführbarer Prozess

```
<partnerLinks>
  <partnerLink name="purchasing"
    partnerLinkType="ins:purchasingLT" myRole="purchaseService" /> ..</partnerLinks>
```

PartnerLinks (Interfaces)

```
<variables>
  <variable name="PO" messageType="ins:POMessage" /> ... </variables>
```

Variablen (Prozesszustände)

```
<faultHandlers>
  <catch faultName="ins:cannotCompleteOrder"
    faultVariable="POFault"
    faultMessageType="ins:orderFaultType">
    <reply partnerLink="purchasing"
      portType="ins:purchaseOrderPT"
      operation="sendPurchaseOrder" variable="POFault"
      faultName="cannotCompleteOrder" /> </catch> </faultHandlers>
```

Fehlerbehandlung

(Vereinf.) BPEL-Spezifikation am Beispiel (2. Teil)

<sequence>

Sequenz (Anfang bis Ende)

```
<receive partnerLink="purchasing" portType="Ins:purchaseOrderPT"
  operation="sendPurchaseOrder" variable="PO" createInstance="yes">
  <documentation>Receive Purchase Order</documentation>
```

Eintreffen
„PO“

</receive>

<flow>

Parallelität

```
<documentation> A parallel flow (shipping, invoicing, scheduling )</documentation>
```

<links>

```
<link name="ship-to-invoice" />
<link name="ship-to-scheduling" /> </links>
```

Ausführungsreihenfolge
durch Abhängigkeiten

```
<sequence> ..... </ sequence>
```

```
<sequence> ..... </ sequence>
```

```
<sequence> ..... </ sequence>
```

3 Sequenzen: „invoicing,
shipping, scheduling“

</flow>

```
<reply partnerLink="purchasing" portType="Ins:purchaseOrderPT"
  operation="sendPurchaseOrder" variable="Invoice">
  <documentation>Invoice Processing</documentation>
```

Senden „Invoice“

</reply>

</ sequence>

</ process>

Preisberechnung in BPEL am Beispiel

```
<sequence>
```

```
<invoke partnerLink="invoicing"  
  portType="Ins:computePricePT"  
  operation="initiatePriceCalculation"  
  inputVariable="PO">
```

Web Service Aufruf

```
<documentation> Initial Price Calculation </documentation>
```

```
</invoke>
```

```
<invoke partnerLink="invoicing"  
  portType="Ins:computePricePT"  
  operation="sendShippingPrice"  
  inputVariable="shippingInfo">
```

Web Service Aufruf

Variable

```
<documentation> Complete Price Calculation </documentation>
```

```
<targets>
```

```
<target linkName="ship-to-invoice" />
```

Link

```
</targets>
```

```
</invoke>
```

```
<receive partnerLink="invoicing"  
  portType="Ins:invoiceCallbackPT"  
  operation="sendInvoice" variable="Invoice" />
```

Nachrichteneingang

```
</sequence>
```

Sprachkonstrukte BPEL (I)

WS-BPEL

Historie

Überblick

Beispiel

Sprachkonstrukte

Umsetzung

Mapping

- ◆ Top-Level: `<process> </process>`
 - Attribute (von Element `<process>`): Prozessname, Namespace, abstrakter oder ausführbarer Prozess
 - Abstrakter Prozess: externe Verhalten des Prozesses
 - Ausführbarer Prozess: beschreibt die interne Implementierung
- ◆ Elemente
 - `<partnerLinks>` Identifikation von externen Web Services, die aufgerufen werden
 - `<variables>` Datenfluss des Prozesses
 - `<correlation sets>` Binden einer Menge von Operationen zu einer Service Instanz
 - `<fault Handlers>` Ausnahme-Behandlung
 - `<compensation Handlers>` Aktionen für Transaktions-Rollback
 - `<event Handlers>` Aktionen als Antwort auf externe Events

Sprachkonstrukte BPEL (II)

- ◆ Prozesslogik
 - Menge von BPEL-Aktivitäten, die strukturiert ausgeführt werden
 - Kontrollstrukturen vergleichsweise wie bei Programmiersprachen
 - „Aufgaben“ eines Prozesses (Terminologie Modellierung: Aktivitäten) werden durch das Aufrufen von Web Services ausgeführt

Aktivitäten in BPEL

- ◆ Unterscheidung
 - Einfache Aktivitäten („Basic Activities“)
 - Z.B. Kommunikation mit Partnern, Zuweisungen, Fehlerbehandlung, Prozessende
 - Strukturierte Aktivitäten („Structured Activities“)
 - Strukturierung des Prozessablaufes
 - Z.B. Parallele Ausführung, Schleifen, Bedingungen, Ereignisbedingte Ausführungen

Einfache Aktivitäten in BPEL

- ◆ <invoke>
 - Aufruf einer Operation, die von einem Web Service zur Verfügung gestellt wird
- ◆ <receive>
 - Empfang einer Nachricht eines Web Service Aufrufes
- ◆ <reply>
 - Antwort an Web Service auf gesendete Nachricht
- ◆ <assign>
 - Zuordnung von Werten (Variablen)
- ◆ <throw>
 - signalisiert Fehler („Exception-Handling“)
- ◆ <wait>
 - warten auf Deadline oder Zeitablauf
- ◆ <empty>
 - keine Aktion (z.B. als Synchronisations-Punkt)
- ◆ <exit>
 - sofortiger Abbruch einer Workflow Instanz
- ◆ <rethrow>
 - (in Kombination mit <catch> und <catchAll>)

Strukturierte Aktivitäten in BPEL

- ◆ <sequence>
 - sequentielle Ausführung von Aktivitäten
- ◆ <if>
 - konditionale Verzweigungen durch Verknüpfung von Bedingungen (<condition>)
- ◆ <while>, <repeatUntil>
 - wiederholtes Ausführen einer Aktivität
- ◆ <pick>
 - Triggern einer Aktivität als Reaktion auf eintreffendes Ereignis
- ◆ <flow>
 - parallele Verarbeitung von Aktivitäten
 - <link> Konstrukt innerhalb eines <flow> Konstruktes: explizite Synchronisation zwischen verschachtelten „Kinder“-Aktivitäten
- ◆ <forEach>
 - Mehrfache Ausführung einer Aktivität (parallel oder sequentiell)

(Einige) weitere Konstrukte in BPEL

- ◆ `<scope>`
 - Definiert einen lokalen Kontext mit eigenen Variablen, Partner-Links, Event-Handlern, etc.
 - Hierarchische Strukturierung von „scopes“ möglich
 - Beispiel: `<scope name="Scope" />`
- ◆ `<link>`
 - Ausführungs-Constraints (Synchronisation) zwischen Aktivitäten
 - `<source>`, `<target>`
- ◆ `<compensationHandler>`
 - Kompensationsmechanismus (Transaktionen)
- ◆ `<terminate>`
 - Ende des Prozesses

Umsetzung von BPEL

WS-BPEL

Historie

Überblick

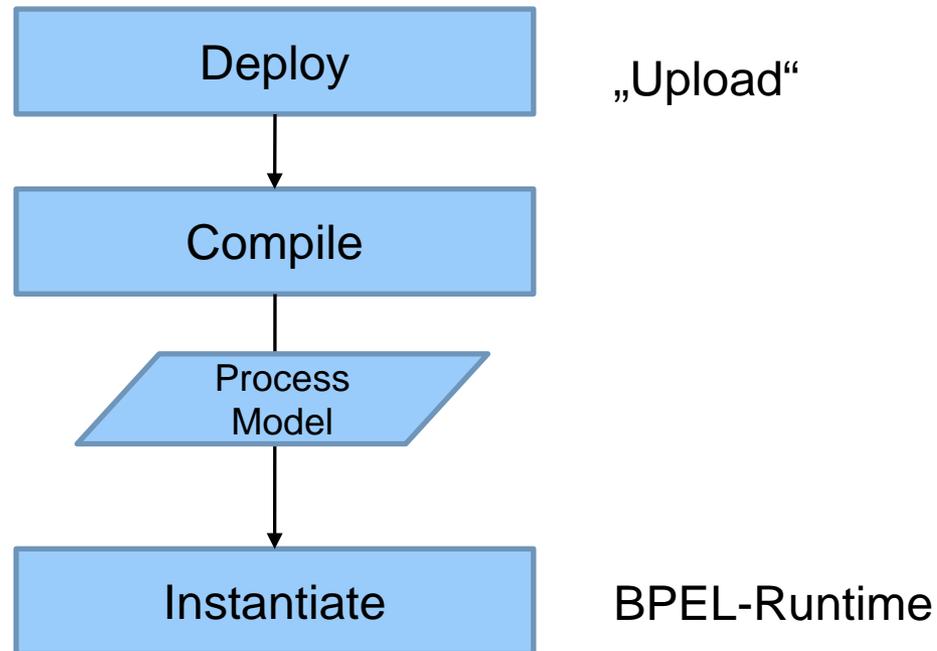
Beispiel

Sprachkonstrukte

Umsetzung

Mapping

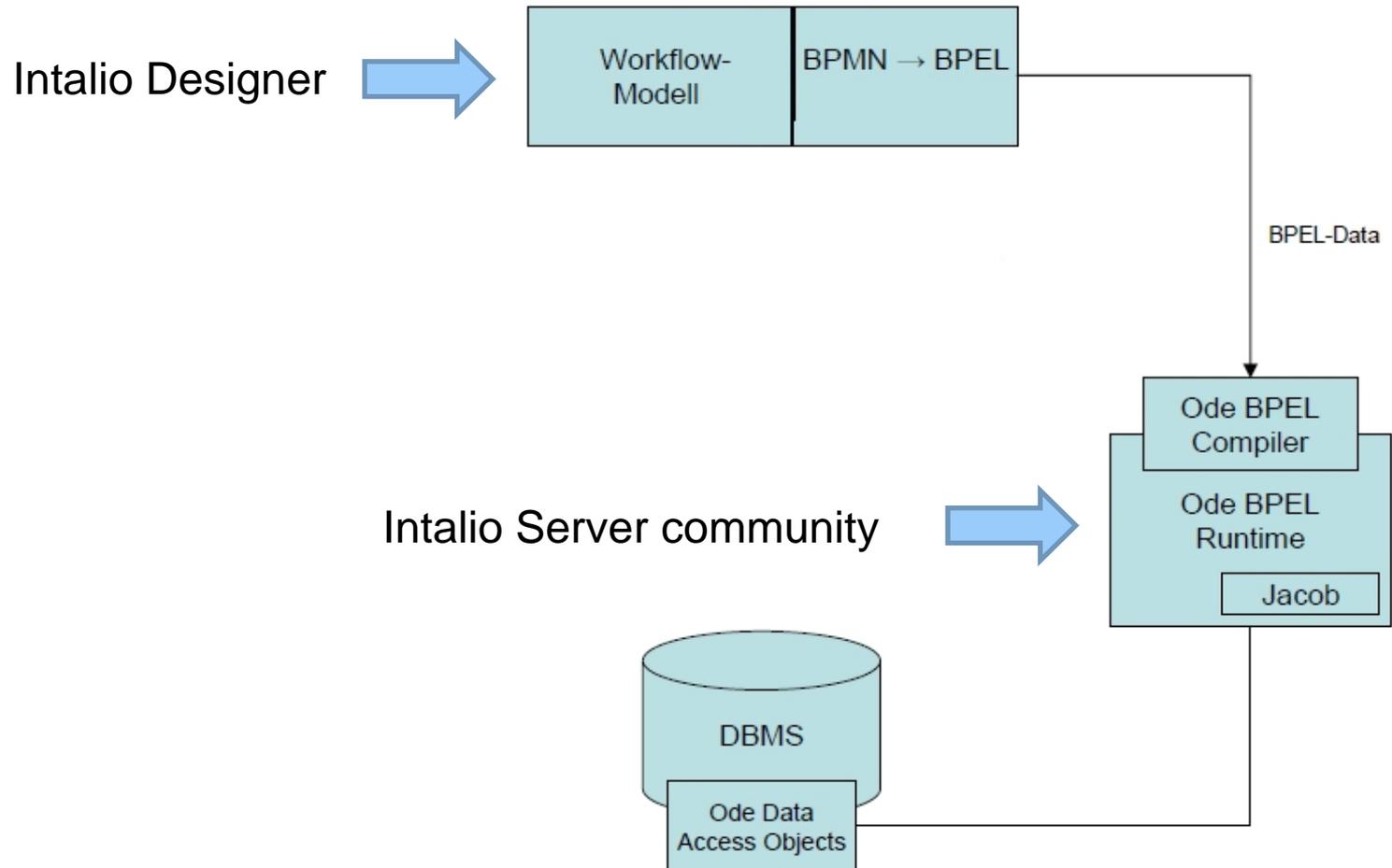
- ◆ Ausführungsumgebung: sogenannte BPEL-Engine



Beispiele BPEL-Engines

- ◆ Oracle BPEL Process Manager
- ◆ IBM WebSphere Process Server
- ◆ Microsoft BizTalk Server
- ◆ Apache ODE (Open Source)
- ◆ JBoss jBPM
- ◆ Intalio BPMS (basiert auf Apache ODE)

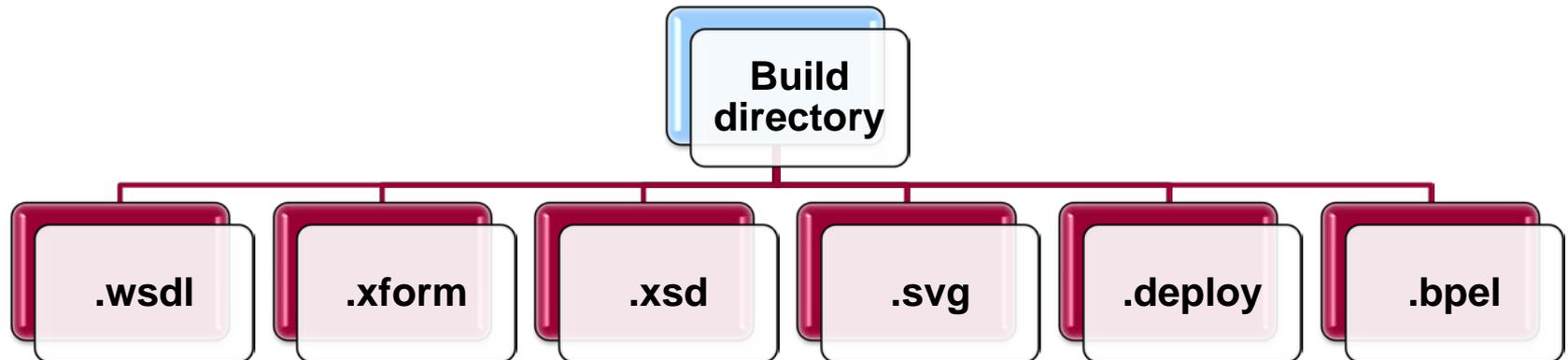
Transformation am Beispiel Intalio



Intalio BPMS Designer

Erzeugung folgender Dateien:

- *.wsdl*
- *.xsd*
- *.xform*
- *.svg*
- *.deploy*
- *.bpel*



Mapping von BPMN nach BPEL

WS-BPEL

Historie

Überblick

Beispiel

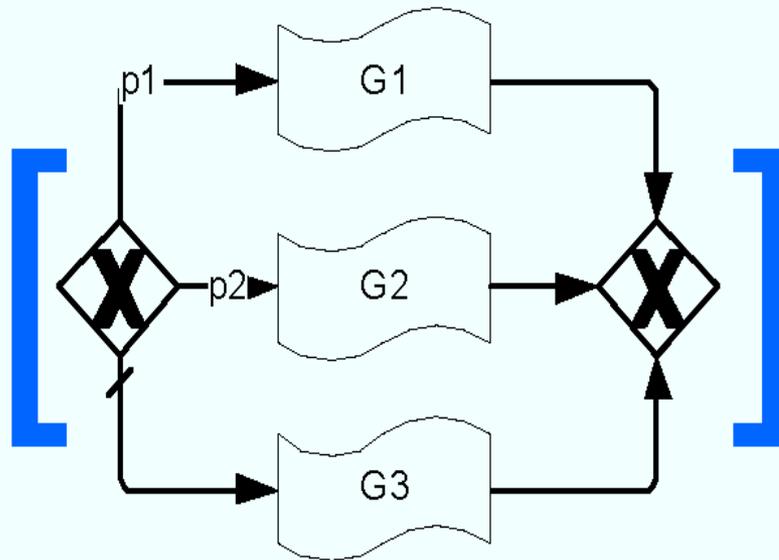
Sprachkonstrukte

Umsetzung

Mapping

- ◆ Erzeugung von BPEL üblicherweise automatisch (z.B. mit Hilfe eines graph. Modellierungswerkzeuges von BPMN nach BPEL)
- ◆ Problematik: unterschiedliche Ausdrucksmächtigkeit der Sprachen
 - Z.B. Datenaspekt bei BPMN vernachlässigt
- ◆ BPMN Standard: Vorschlag zur Transformation von BPMN 2.0 nach BPEL 2.0

Beispiel Abbildung BPMN -> BPEL



=

```
<if><condition>[p1]</condition>
  [G1]
<elseif><condition>[p2]</condition>
  [G2]
</elseif>
<else>
  [G3]
</else>
</if>
```

Quelle: BPMN 2.0 Standard

Exemplarische Fragen zu Kapitel 8

- ◆ Wodurch unterscheiden sich BPMN und BPEL?
- ◆ Welche Rolle spielen Web Services bei BPEL?
- ◆ Nennen und beschreiben Sie kurz drei einfache und drei strukturierte Aktivitäten in BPEL.
- ◆ Beschreiben Sie den sequentiellen und parallelen Aufruf von zwei Web Services mit Hilfe von BPEL-Aktivitäten.
- ◆ Wie wird der Datenfluss in BPEL dargestellt?
- ◆ Welche Rollen spielen PartnerLinks und Links in BPEL?

Ergänzende Literatur zu Kapitel 8

- ◆ WS-BPEL Standard
<http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.html>
- ◆ Weske: Business Process Management: Concepts, Languages, Architectures. Springer 2007
- ◆ BPMN 2.0 Standard (Kap. 14): BPMN -> BPEL
- ◆ www.intalio.com