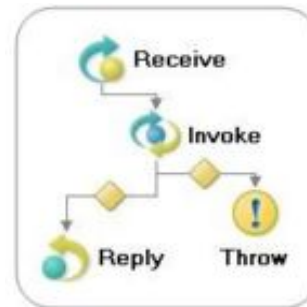


# Daten- und Systemintegration

## Business Process Execution Language - v2.0



Prof. Dr. Frank Bühler  
Prof. Dr. Günter Turetschek

# Vorlesung „ Daten- und Systemintegration“

## **Kapitel 8      Business Prozess Management**

- 8.1      **BPEL (Business Process Execution Language)**
- 8.2      BPEL Server-Produkte
- 8.3      BPEL4PEOPLE
- 8.4      BPELJ
- 8.5      Transformation eEPK->BPEL, BPMN->BPEL
- 8.6      Links+Artikel

# 8.1 Business Process Execution Language

## Ereignisgetriebene Anwendungen und Prozesse

Die Geschäftsprozesse werden durch Ereignisse getrieben, z. B.

- + eine Bestellung trifft ein
- + eine Ware wird bestellt
- + eine Maschine fällt aus
- + eine Person wird angestellt

Die Prozesse müssen sich an diesen Ereignissen ausrichten.

Sofern Probleme auftreten, die ein Eingreifen bestimmter Personen erfordert, dann müssen diese umgehend benachrichtet werden.

Damit dies funktioniert, müssen bestehende Systeme angepasst und neue Systeme entsprechend entworfen werden.

# 8.1 Business Process Execution Language

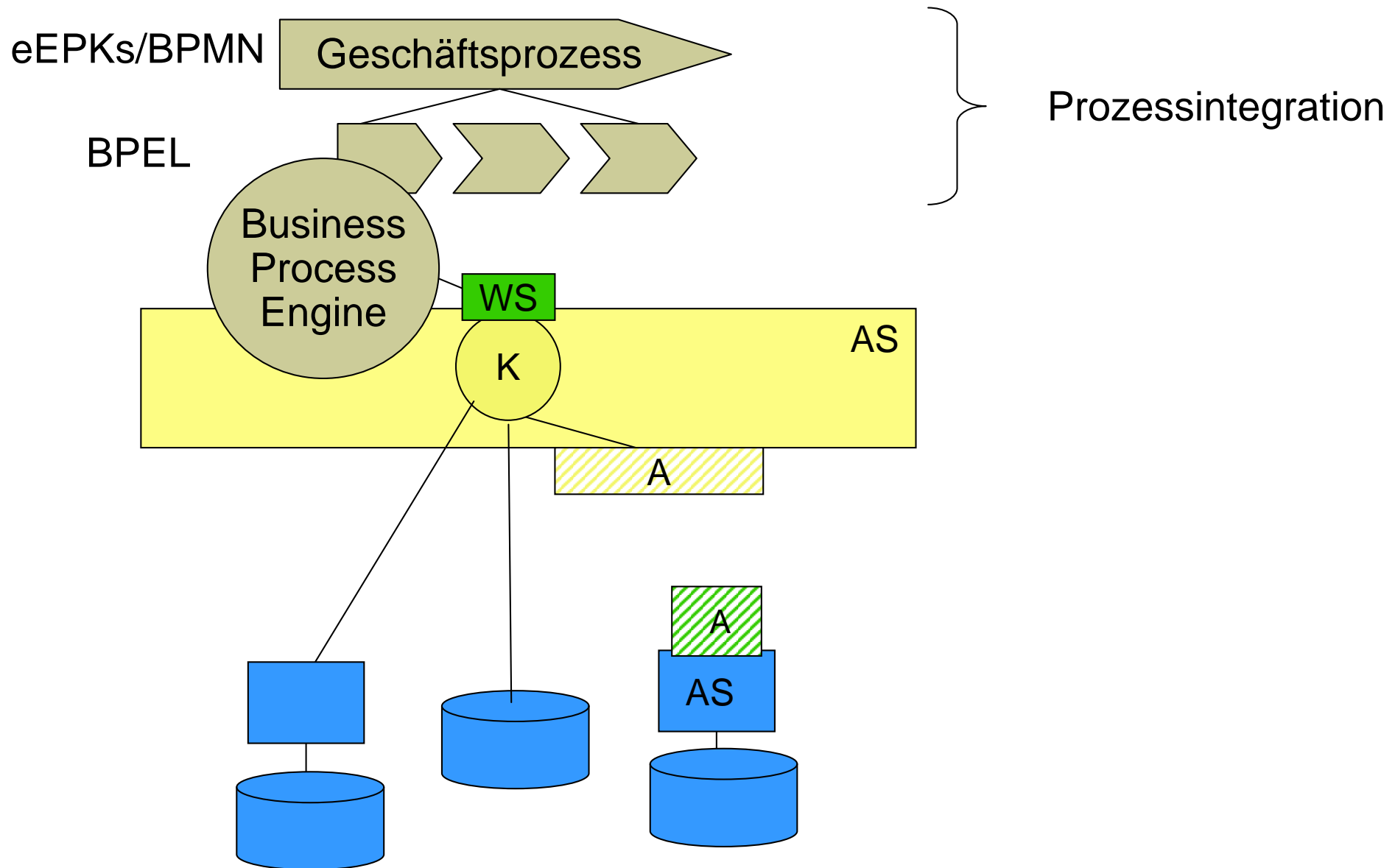
## Ereignisgetriebene Anwendungen und Prozesse

Der Einsatz von **eEPKS** und **BPMN-Prozessdiagramme** sind gut hierfür geeignet, um dieses Paradigma sowohl auf der strategischen als auch operationalen Ebene zu unterstützen.

Ereignisse lösen die verschiedenen **Prozessschritte** aus, die manuell oder maschinell umgesetzt werden können. Zur Ausführung der verschiedenen **maschinellen Aktivitäten** eignen sich insbesondere Service-orientierte Architekturen (SOA) und **WebServices**.

Auf der technischen Seite helfen Sprachen wie beispielsweise **WS-BPEL**. Diese nutzen verfügbare Systemfunktionen, um Abläufe für Geschäftsprozesse umzusetzen.

# 8.1 Business Process Execution Language



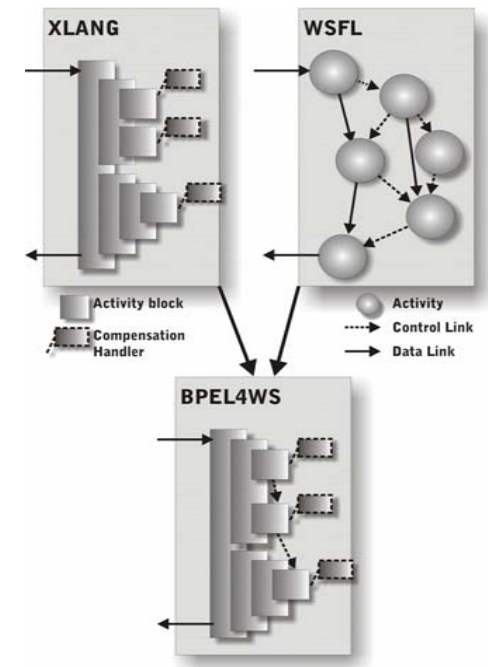
# 8.1 Business Process Execution Language

## Motivation für WS-BPEL

**WSDL reicht nicht aus, da Web Services zustandslos sind.**

### Anforderungen an WS-BPEL

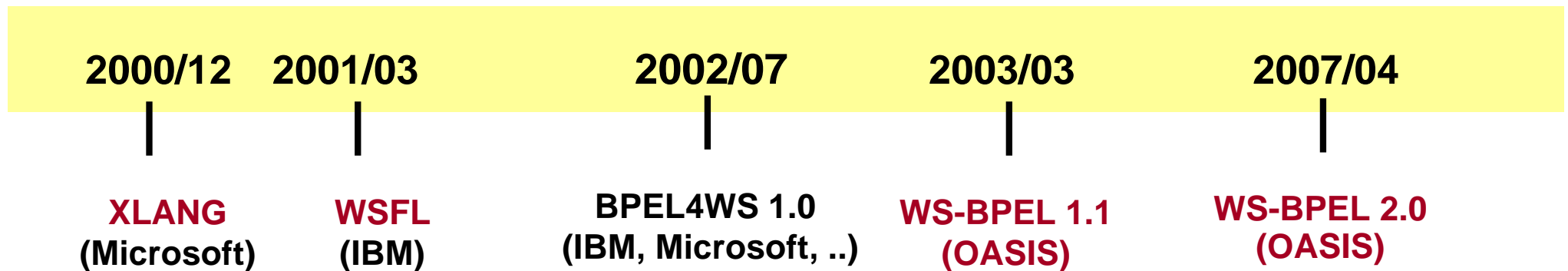
- Synchroner und asynchroner Austausch von Nachrichten in einer geordneten Folge
- Zustandsbehaftete und langlebige Kommunikations-Verbindung
- Plattformunabhängigkeit
- Unterstützung von WebServices



# 8.1 Business Process Execution Language

## Entwicklungsgeschichte

## Business Process Execution Language (BPEL)



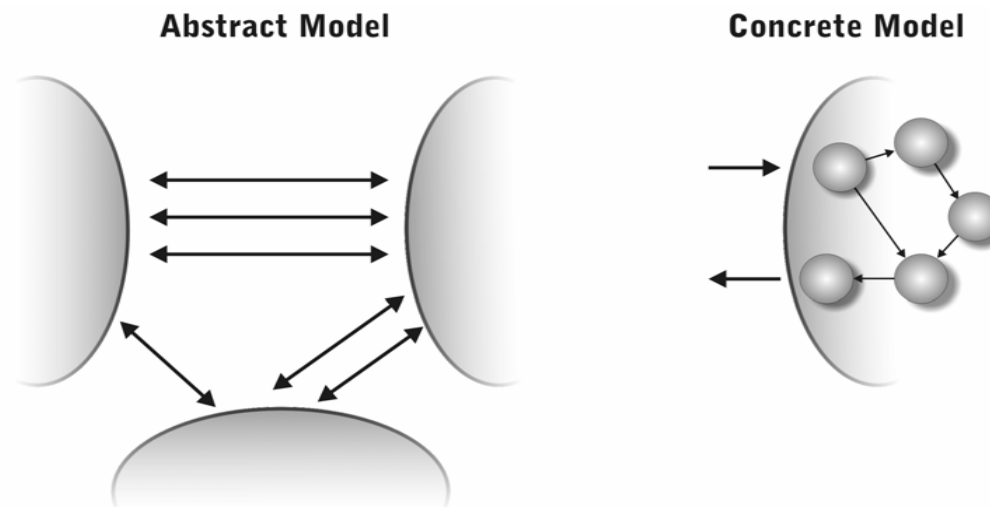
BPEL 2.0 siehe [http://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=wsbpel](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel)

# 8.1 Business Process Execution Language

## Business Process Execution Language for Web Services (WS-BPEL)

### WS-BPEL unterstützt zwei verschiedene Szenarien

- Implementierung *konkreter*, ausführbarer Geschäftsprozesse innerhalb eines Unternehmens ("Orchestrierung")
- Beschreibung *abstrakter*, nicht ausführbarer Prozesse zwischen zwei Partnern ("Choreographie")



# 8.1 Business Process Execution Language

## “Orchestration” vs “Choreography”

### **Orchestrierung**

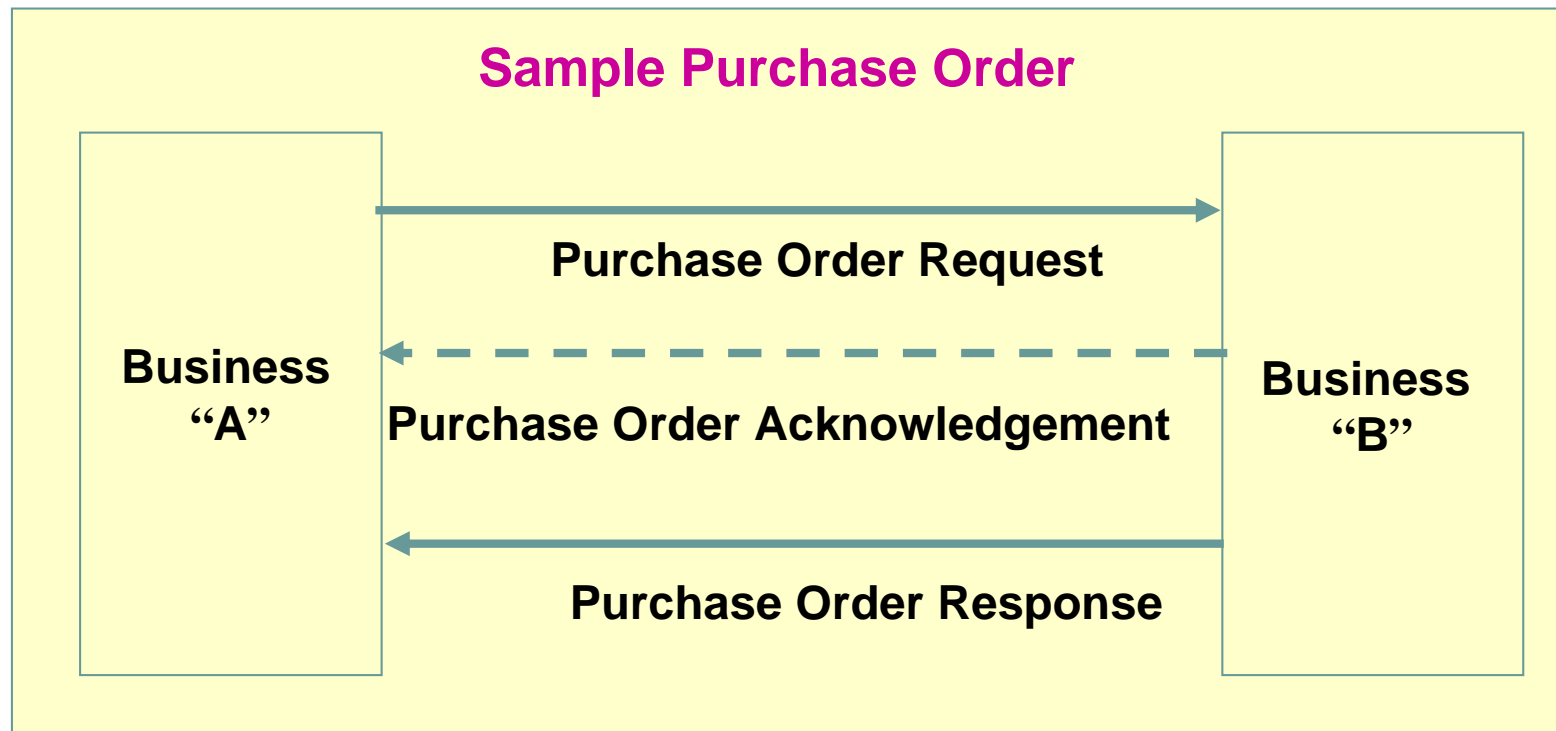
Orchestrierung beschreibt einen ausführbaren Geschäftsprozess von der Perspektive und Kontrolle eines einzelnen Endpunktes. Dies entspricht einem Workflow innerhalb eines Unternehmens.

### **Choreographie**

Der beobachtbare öffentliche Austausch von Nachrichten oder Vereinbarungen zwischen zwei oder mehr Partnern. Hierunter fallen Sprachen wie WS-BPEL.

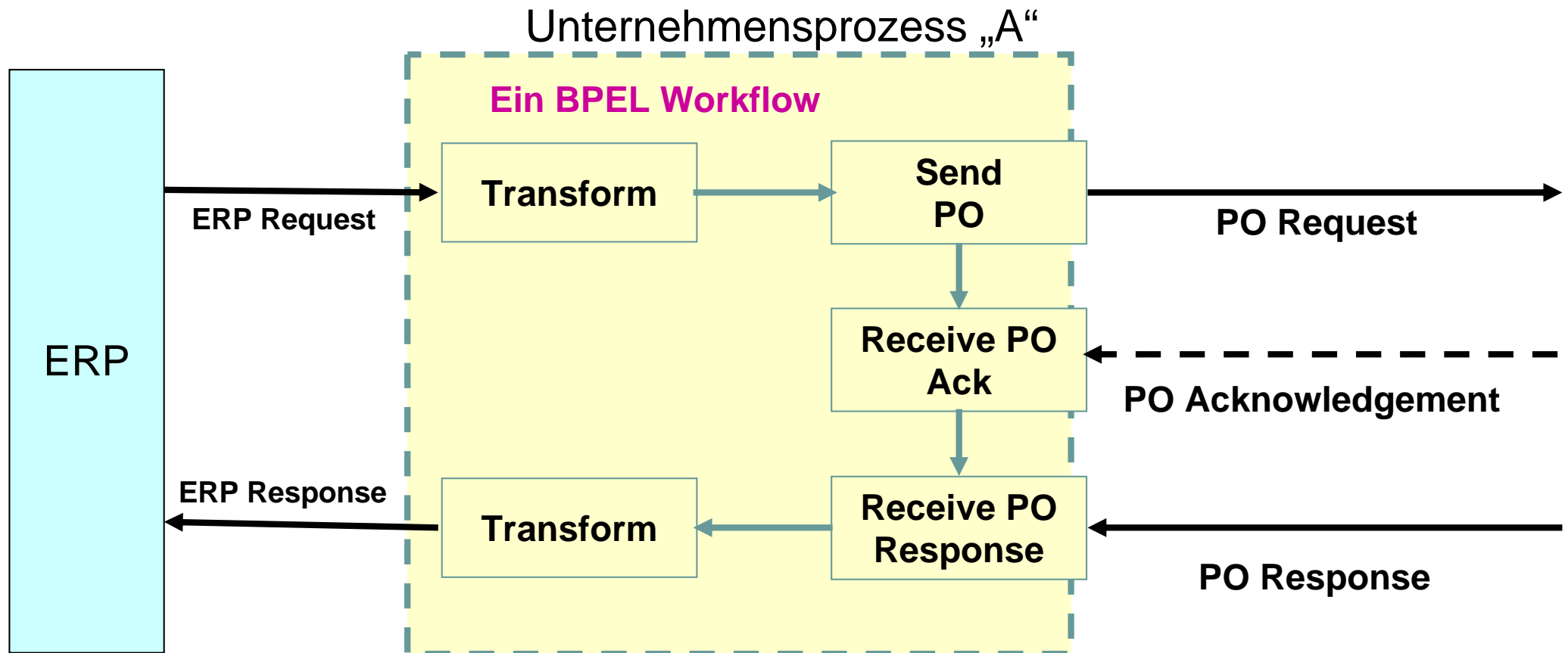
# 8.1 Business Process Execution Language

## Beispiel eines B2B-Geschäftsprozesses: Purchase Order



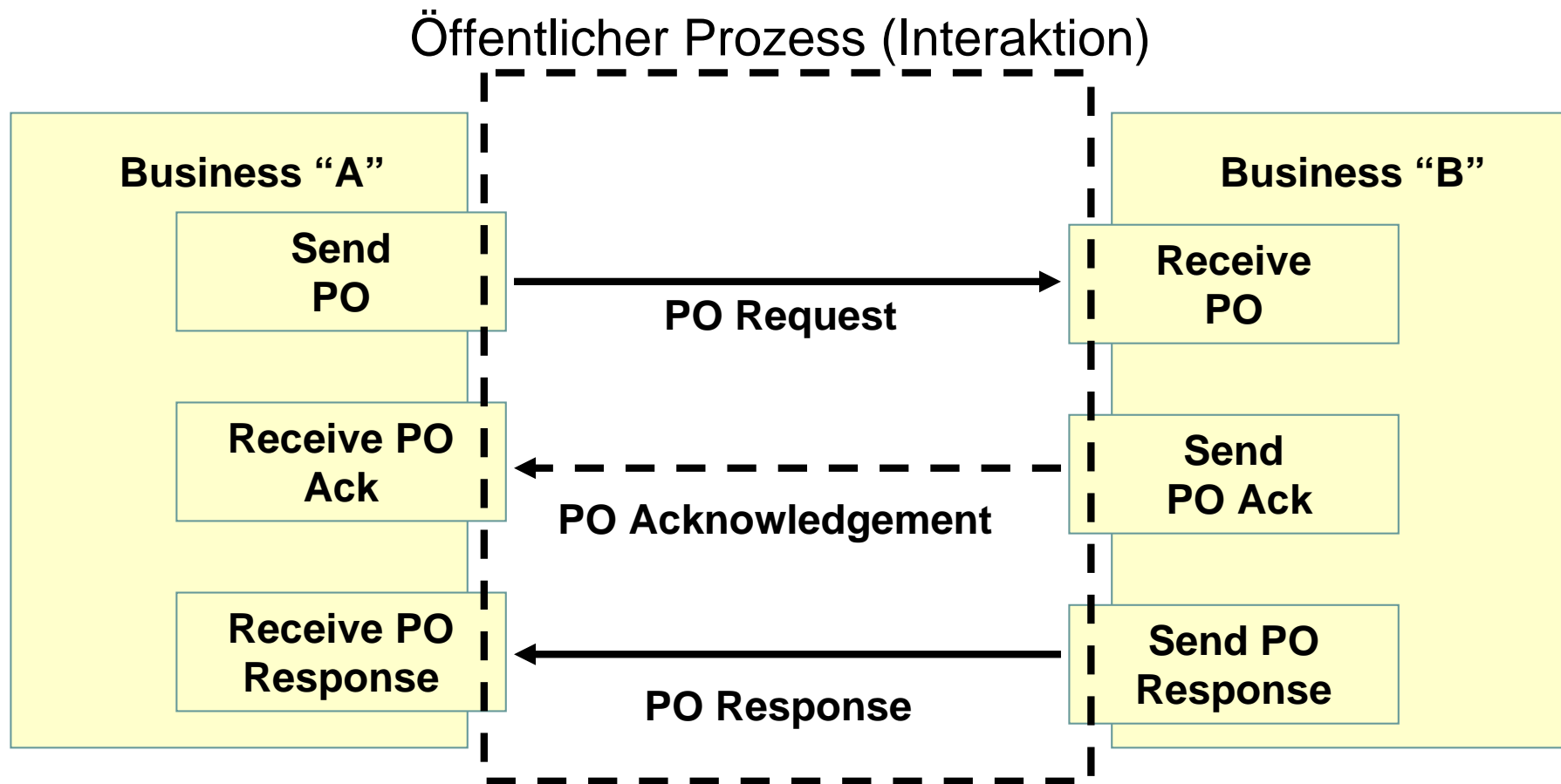
# 8.1 Business Process Execution Language

## Orchestrierungs-Perspektive



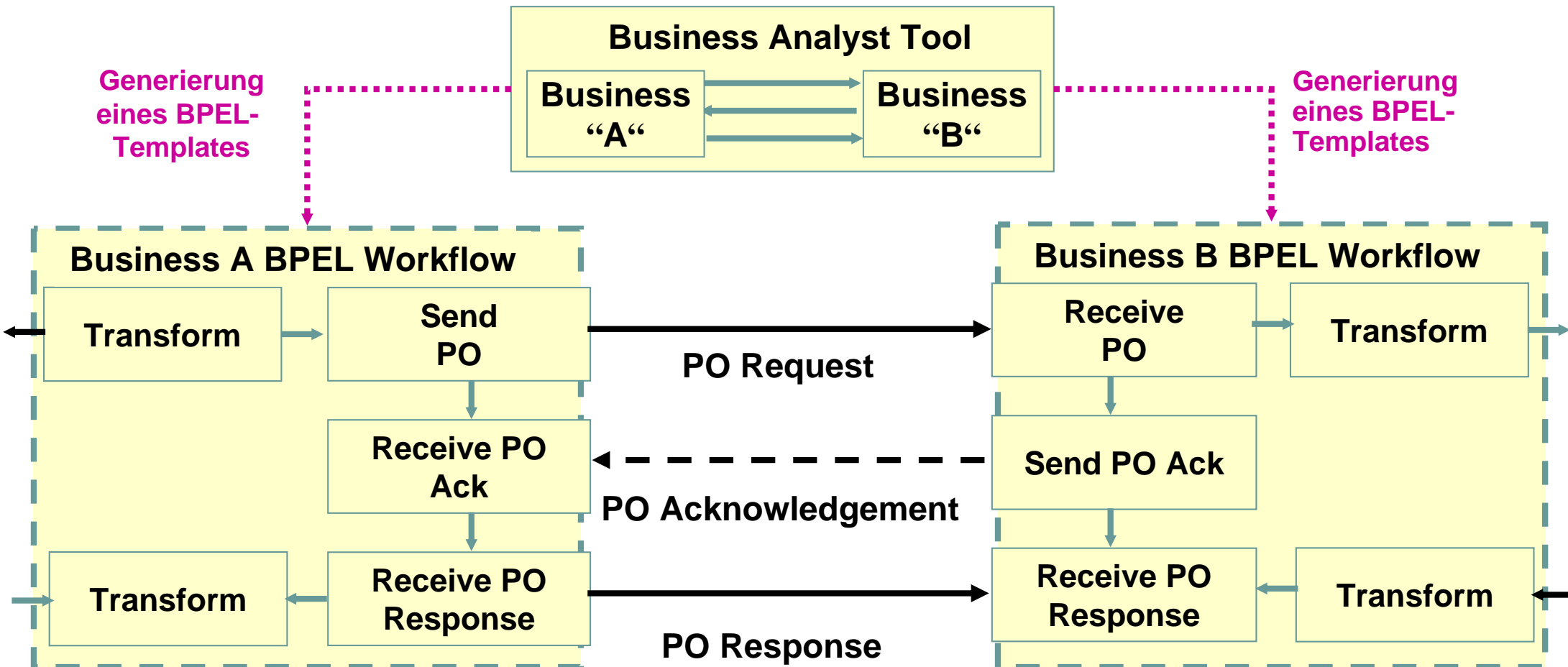
# 8.1 Business Process Execution Language

## Choreographie-Perspektive



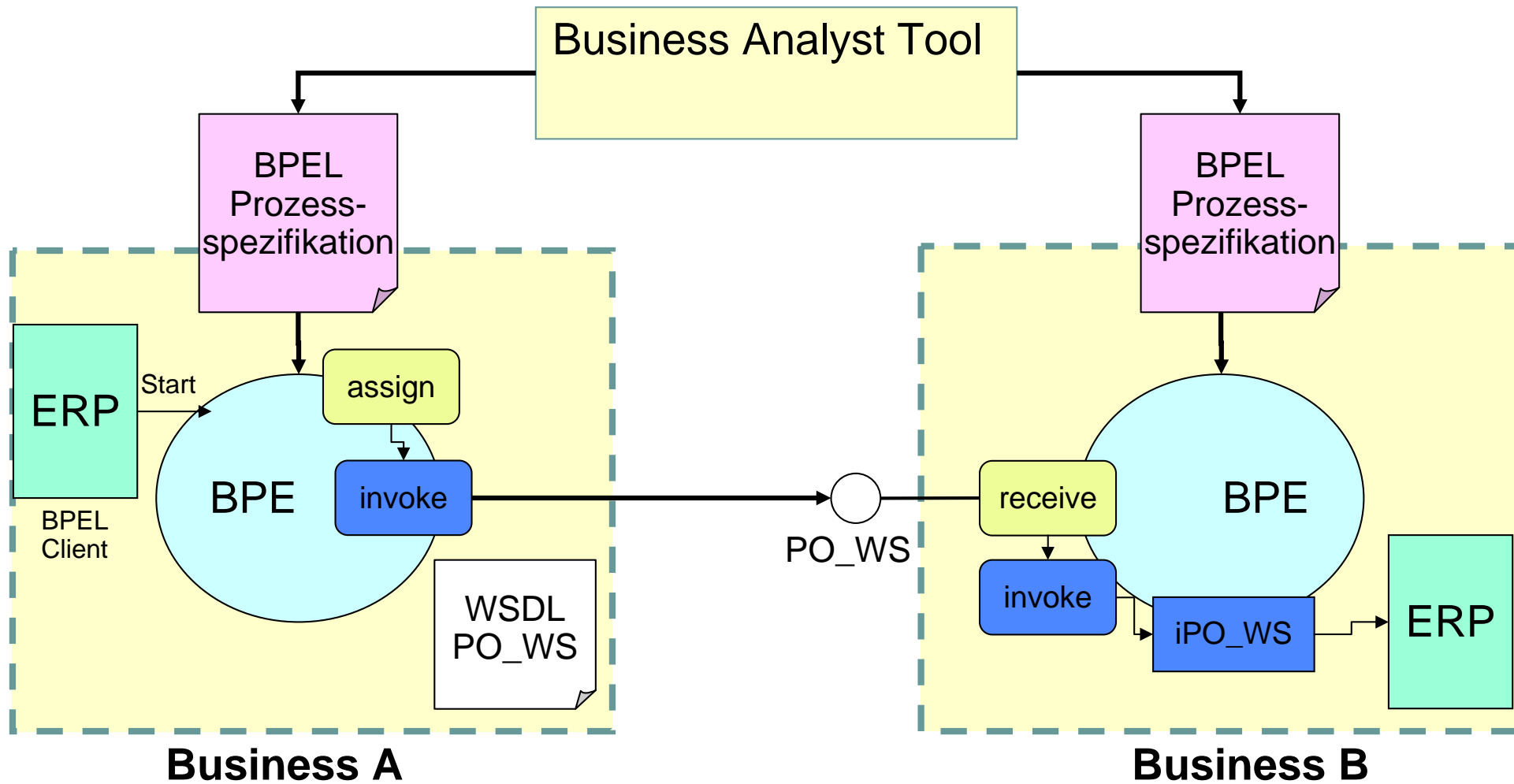
# 8.1 Business Process Execution Language

## Orchestrierung und Choreographie auf einen Blick



# 8.1 Business Process Execution Language

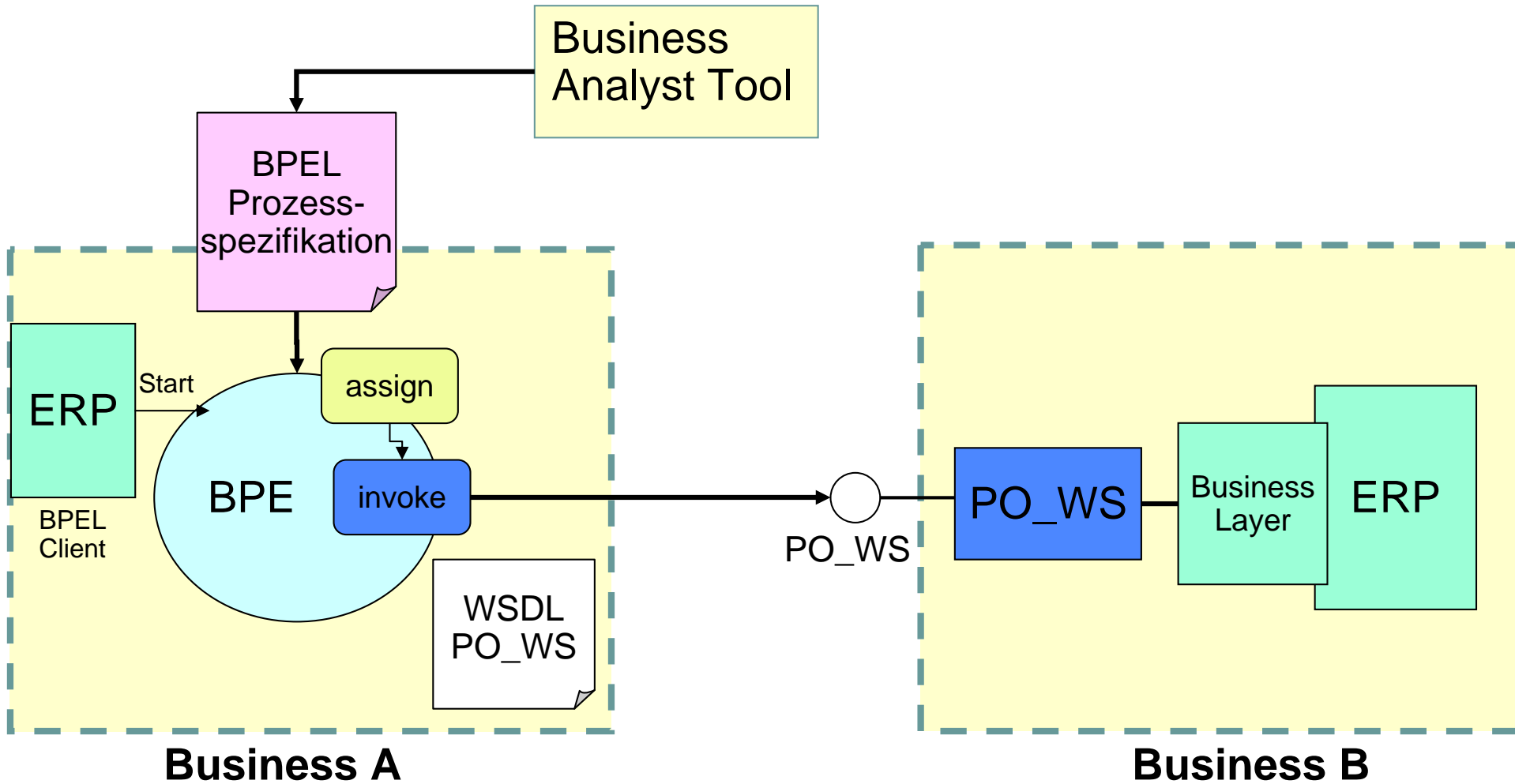
## Umsetzung mittels BPEL – Version 1



BPE = Business Process Engine

# 8.1 Business Process Execution Language

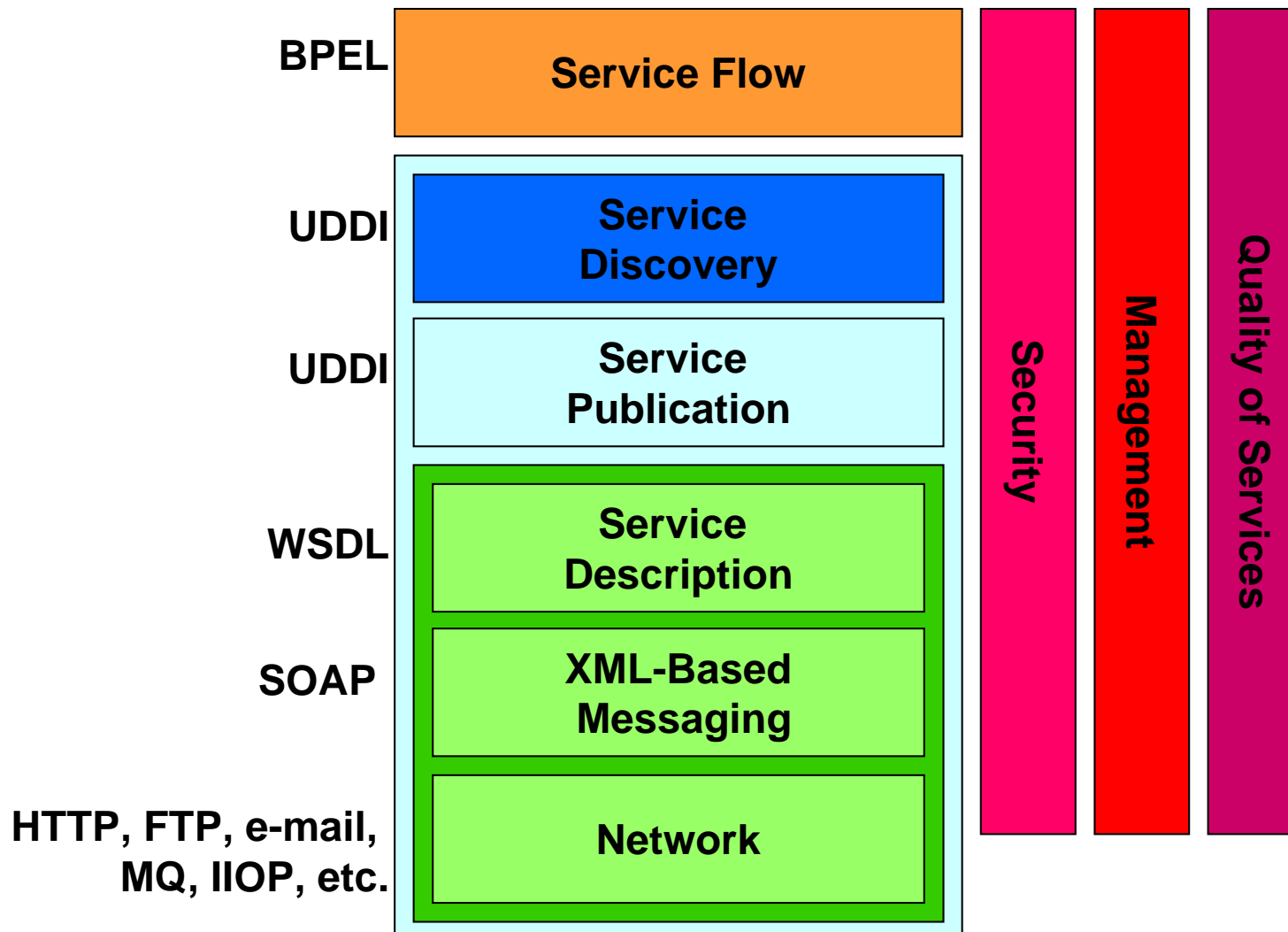
## Umsetzung mittels BPEL – Version 2



BPE = Business Process Engine

# 8.1 Business Process Execution Language

## Einordnung von BPEL – 1(2)



Die "Business Process Execution Language" (BPEL) ermöglicht die Beschreibung von Abläufen auf Basis von Web Services.

Es wird die Interaktion aller beteiligter Partner festgelegt.

# 8.1 Business Process Execution Language

## Einordnung von BPEL – 2(2)

Management	Choreography – WS-CDL (W3C)			Business Processes
	<b>Orchestration - WS-BPEL (OASIS)</b>			
	WS-Reliability	WS-Security	Transactions	Quality of Service
			Coordination	
			Context	
	UDDI			Discovery
	WSDL			Description
	SOAP			Message
	XML			
	HTTP, IIOP, JMS, SMTP			Transport

# 8.1 Business Process Execution Language

## Allg. Vorgehen bzgl. Implementierung eines WS-BPEL Workflows

- Erstellen eines WS-BPEL-Workflow
- Zusammenstellung aller benötigter WSDL-Dokumente
- Verteilung des WS-BPEL-Workflows sowie der WSDL-Dokumente auf die Ausführungs-Engine
- Ausführung des Workflows durch Business Process Engine ("BPEL-Console")

# 8.1 Business Process Execution Language

## **Business Process Execution Language for Web Services (WS-BPEL)**

Version 1.0 IBM, Microsoft und BEA (August 2002)

Version 1.1 OASIS (April 2003)

Version 2.0 OASIS (2007)

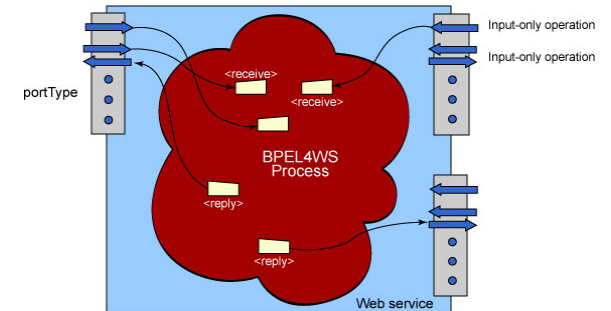
WS-BPEL ermöglicht auf Basis von WebServices die Spezifikation von komplexen, ausführbaren Prozessen

### **Vorteile**

- Portable Geschäftsprozesse (auf Basis von standardisierten Web Services)
- Anerkannte Spezifikationsprache
- Vielzahl an verfügbaren Prozessmaschinen

# 8.1 Business Process Execution Language

## Allgemeine Struktur von BPEL-Prozessen



```
<process>
  <!-- Definition and roles of process participants -->
  <partnerLinks> ... </partnerLinks>
  <!-- Data/state used within the process -->
  <variables> ... </variables>
  <!-- Properties that enable conversations -->
  <correlationSets> ... </correlationSets>
  <!-- Exception handling -->
  <faultHandlers> ... </faultHandlers>
  <!-- Error recovery - undoing actions -->
  <compensationHandlers> ... </compensationHandlers>
  <!-- Concurrent events with process itself -->
  <eventHandlers> ... </eventHandlers>
  <!-- Business process flow -->
  (activities)*
</process>
```

# 8.1 Business Process Execution Language

## BPEL-Aktivitäten zur Definition des Prozessablaufs

### Primitive Aktivitäten

<invoke>  
<receive>  
<assign>  
<reply>  
<throw>  
<terminate>  
<wait>

### Strukturierte Aktivitäten

<sequence>  
<switch>  
<pick>  
<flow>  
<link>  
<while>  
<scope>

In WS-BPEL 2.0 gibt es syntaktische Verbesserungen/Erweiterungen!

# 8.1 Business Process Execution Language

## Basis-Aktivität "Invoke"

Die bei den Partnern ausführbaren Web Services werden verwendet, um die Geschäftsprozess durchzuführen.

„Invoke“ unterstützt zwei Arten der Kommunikation:

### **Synchroner blockierender Aufruf**

Aufruf- und Rückgabeparameter können verwendet werden.

### **Asynchroner nicht blockierender Aufruf**

Nur Aufrufparameter sind erlaubt.

### **Beispiel**

```
<invoke partner = "ShippingProvider"  
  portType = "shippingPortType"  
  operation = "RequestShipping"  
  inputVariable = "ShippingRequest"  
  outputVariable = "ShippingInfo">  
  
</invoke>
```

# 8.1 Business Process Execution Language

## Basis-Aktivität "Receive"

**Gewöhnlich zu Beginn eines Ablaufs.**

### Beispiel

```
<receive partner = "Customer"  
    portType = "PurchaseOrderPortType"  
    operation = "SendPurchaseOrder"  
    variable = "PO"  
    createInstance = "yes">  
</receive>
```

Verarbeitung des Client-Requests im BPEL-Prozess.

# 8.1 Business Process Execution Language

## Basis-Aktivität "Reply"

**Diese Aktivität ermöglicht die Beantwortung einer eingegangenen Operation. Gewöhnlich die letzte Aktion in einem Ablauf.**

### Beispiel

```
<reply partner = "Customer"  
    portType = "PurchaseOrderPortType"  
    operation = "SendPurchaseOrder"  
    variable = "Invoice">  
</reply>
```

Übermittlung des Ergebnisses des BPEL-Prozesses an den Client.

# 8.1 Business Process Execution Language

## Kontrollstrukturen "Sequence"

**Die einzelnen Aktionen werden nacheinander ausgeführt.**

```
<sequence>  
  <receive> ... </receive>  
  <pick> ... </pick>  
  <invoke> ... </invoke>  
</sequence>
```

# 8.1 Business Process Execution Language

## Kontrollstrukturen "Switch"

### Geordnete Liste von bedingten Zweigen

```
<switch>  
  <case condition= "x = 1">  
    <flow>  
      ...  
    </flow>  
  </case>  
  <case condition= "x = 2">  
    <flow>  
      ...  
    </flow>  
  </case>  
  <otherwise>  
    <throw faultName="Error!"/>  
  </otherwise>  
</switch>
```

# 8.1 Business Process Execution Language

## Kontrollstrukturen "Pick"

**In Abhängigkeit eines Ereignisses wird eine Aktivität ausgeführt.**

```
<pick>  
  <onMessage partner="partner1">  
    <!-- do something -->  
  </onMessage>  
  
  <onMessage partner="partner2">  
    <!-- do something else -->  
  </onMessage>  
  
  <onAlarm for="some_alarm">  
    <!-- handle timeout -->  
  </onAlarm>  
</pick>
```

# 8.1 Business Process Execution Language

## Kontrollstrukturen "Flow"

**Ermöglicht parallele Ausführung von Aktivitäten.**

**Die Flow-Aktivität ist beendet, wenn alle Aktivitäten ausgeführt wurden.**

```
<flow>  
  <invoke partner="partner1" .../>  
  <invoke partner="partner2" .../>  
</flow>
```

## Fork-Join-Logik

# 8.1 Business Process Execution Language

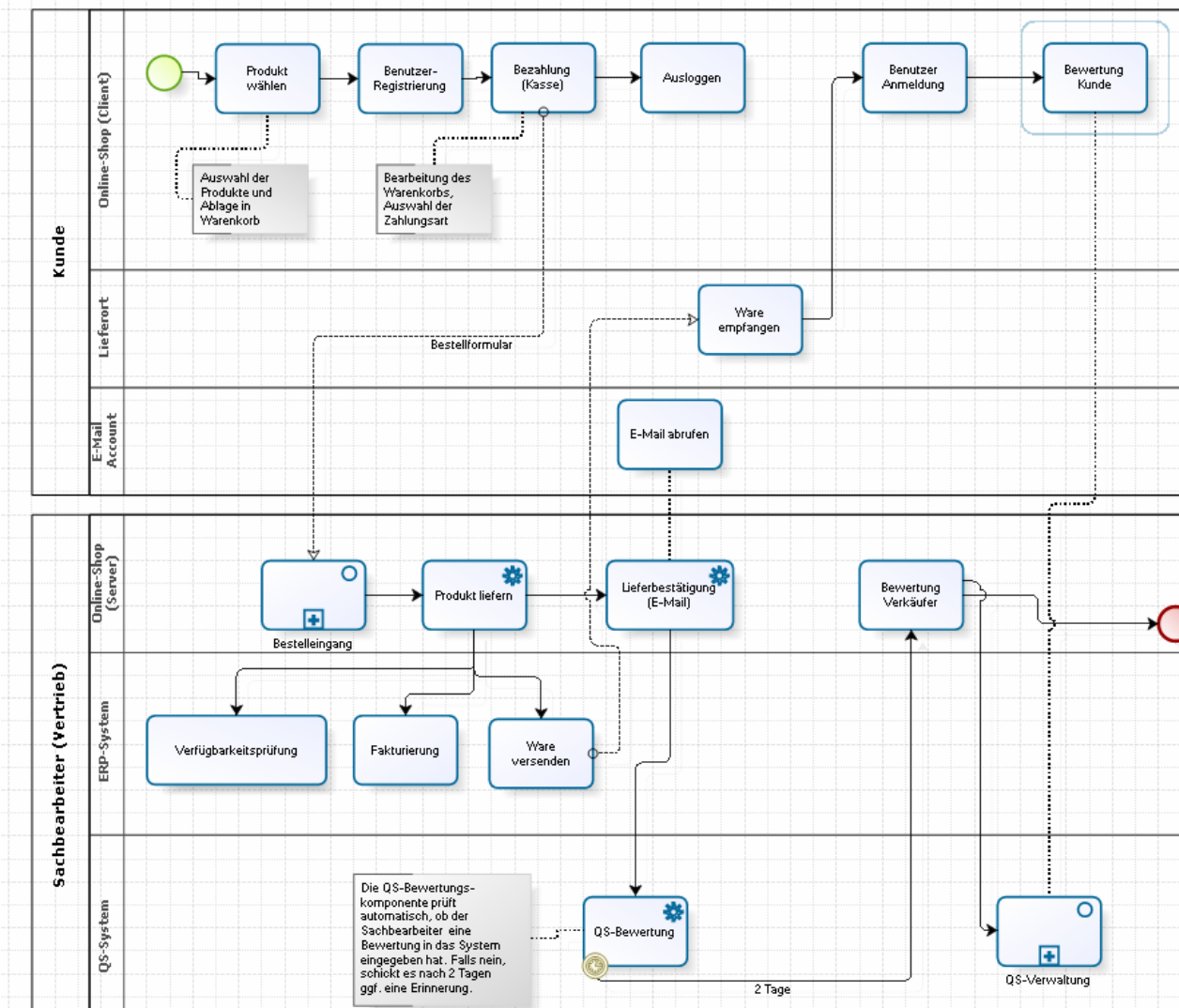
## Kontrollstrukturen "While"

### Iterative Aktivitäten

```
<while condition = "x > 1">  
  <sequence> ... </sequence>  
</while>
```

# 8.1 Business Process Execution Language

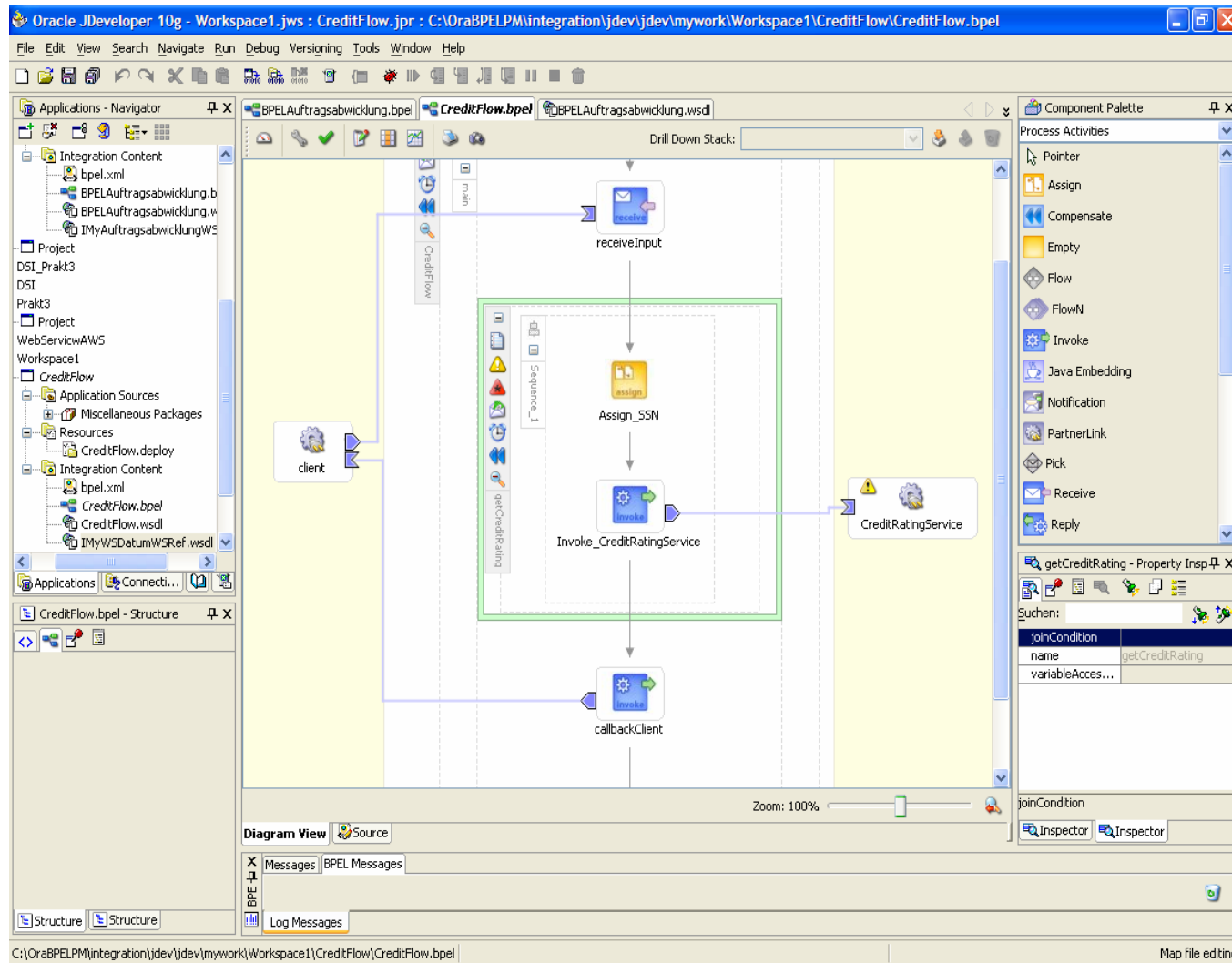
## Beispiel (siehe Hörsaalübung)



Wie sieht die Service-orientierte Modellierung des Prozesses in BPEL aus?

# 8.1 Business Process Execution Language

## Beispiel: CreditFlow (JDeveloper)



Quelle: Oracle QuickStart-Tutorial

# 8.1 Business Process Execution Language

## Beispiel "CreditFlow"-Code – 1(3)

```
<!-- ////////////////////////////////////////////////////////////////////
// Oracle JDeveloper BPEL Designer
// Created: Mon Nov 13 18:30:46 CET 2006
// Author: Prof. Dr. Frank Bühler / Oracle Qick Start Tutorial
// Purpose: Asynchronous BPEL Process
////////////////////////////////////////////////////////////////// -->
<process name="CreditFlow" targetNamespace="http://tutorial.oracle.com"
  xmlns="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
  xmlns:xp20="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functions.Xpath20"
  xmlns:bpws="http://schemas.xmlsoap.org/ws/2003/03/business-process/" xmlns:ns1="http://services.otn.com"
  xmlns:ldap="http://schemas.oracle.com/xpath/extension/ldap" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:ns2="http://mypackage/WSDatum.wsdl" xmlns:client="http://tutorial.oracle.com"
  xmlns:bpelx="http://schemas.oracle.com/bpel/extension" xmlns:ora="http://schemas.oracle.com/xpath/extension"
  xmlns:orcl="http://www.oracle.com/XSL/Transform/java/oracle.tip.pc.services.functions.ExtFunc">

<!-- ===== -->
<!-- PARTNERLINKS -->
<!-- List of services participating in this BPEL process -->
<!-- ===== -->
<partnerLinks><!--
  The 'client' role represents the requester of this service. It is
  used for callback. The location and correlation information associated
  with the client role are automatically set using WS-Addressing.
  -->
  <partnerLink name="client" partnerLinkType="client:CreditFlow" myRole="CreditFlowProvider"
    partnerRole="CreditFlowRequester"/>
  <partnerLink name="CreditRatingService" partnerRole="CreditRatingServiceProvider"
    partnerLinkType="ns1:CreditRatingService"/>
</partnerLinks>
```

# 8.1 Business Process Execution Language

## Beispiel "CreditFlow"-Code – 2(3)

```
<!-- VARIABLES -->
<!-- List of messages and XML documents used within this BPEL process -->
<!-- ===== -->
<variables>
  <!-- Reference to the message passed as input during initiation -->
    <variable name="inputVariable" messageType="client:CreditFlowRequestMessage"/>
  <!-- Reference to the message that will be sent back to the
    requester during callback
    -->
    <variable name="outputVariable" messageType="client:CreditFlowResponseMessage"/>
    <variable name="crInput" messageType="ns1:CreditRatingServiceRequestMessage"/>
    <variable name="crOutput" messageType="ns1:CreditRatingServiceResponseMessage"/>
</variables>
```

# 8.1 Business Process Execution Language

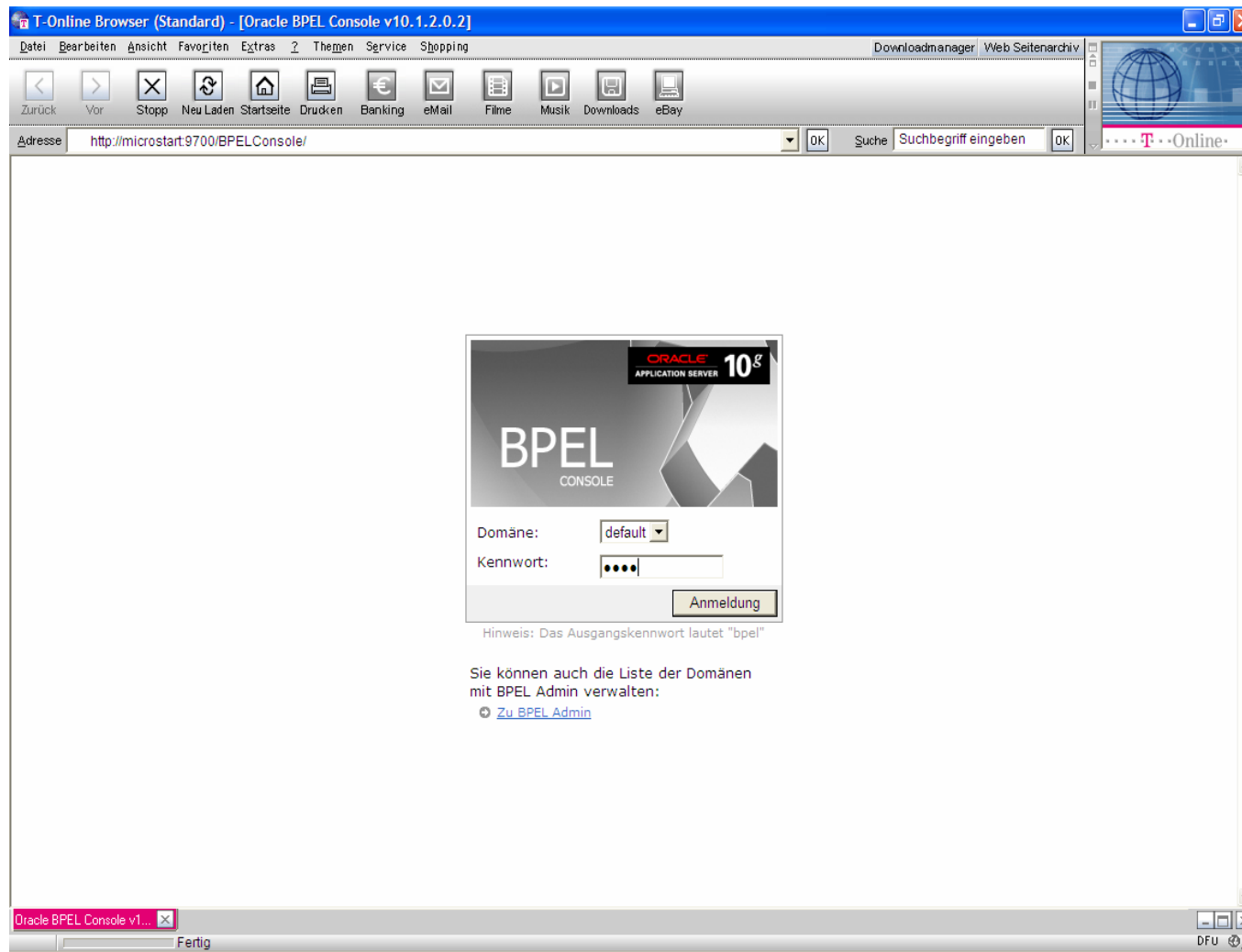
## Beispiel "CreditFlow"-Code – 3(3)

```
<!-- ORCHESTRATION LOGIC -->
<!-- Set of activities coordinating the flow of messages across the -->
<!-- services integrated within this business process -->
<!-- ===== -->
<sequence name="main"><!-- Receive input from requestor.
Note: This maps to operation defined in CreditFlow.wsdl -->
  <receive name="receiveInput" partnerLink="client" portType="client:CreditFlow" operation="initiate"
    variable="inputVariable" createInstance="yes"/>

  <scope name="getCreditRating">
    <sequence name="Sequence_1">
      <assign name="Assign_SSN">
        <copy>
          <from variable="inputVariable" part="payload" query="/client:CreditFlowProcessRequest/client:ssn"/>
          <to variable="crInput" part="payload" query="/ns1:ssn"/>
        </copy>
      </assign>
      <invoke name="Invoke_CreditRatingService" partnerLink="CreditRatingService"
        portType="ns1:CreditRatingService" operation="process" inputVariable="crInput" outputVariable="crOutput"/>
    </sequence>
  </scope>
  <invoke name="callbackClient" partnerLink="client" portType="client:CreditFlowCallback" operation="onResult"
    inputVariable="outputVariable"/>
</sequence>
</process>
```

# 8.1 Business Process Execution Language

## Beispiel: CreditFlow (BPEL Console - Anmeldung)



# 8.1 Business Process Execution Language

## Beispiel: CreditFlow (BPEL Console - Dashboard)

The screenshot shows the Oracle BPEL Console interface in a web browser. The browser title is "T-Online Browser (Standard) - [Oracle BPEL Console v10.1.2.0.2]". The address bar shows "http://microstart:9700/BPELConsole/default/index.jsp". The page title is "ORACLE BPEL Console". The navigation tabs are "Dashboard", "BPEL-Prozesse", "Instanzen", and "Aktivitäten". The "Dashboard" tab is active.

The dashboard is divided into two main sections: "Bereitgestellte BPEL-Prozesse" and "Laufende BPEL-Prozessinstanzen".

**Bereitgestellte BPEL-Prozesse:**

Name
BPELAuftragsabwicklung
BPELWSTest1
CreditFlow ( v. 1.0 ) ★
<a href="#">CreditFlow ( v. 2.0 )</a>
CreditRatingService
TaskActionHandler
TaskManager

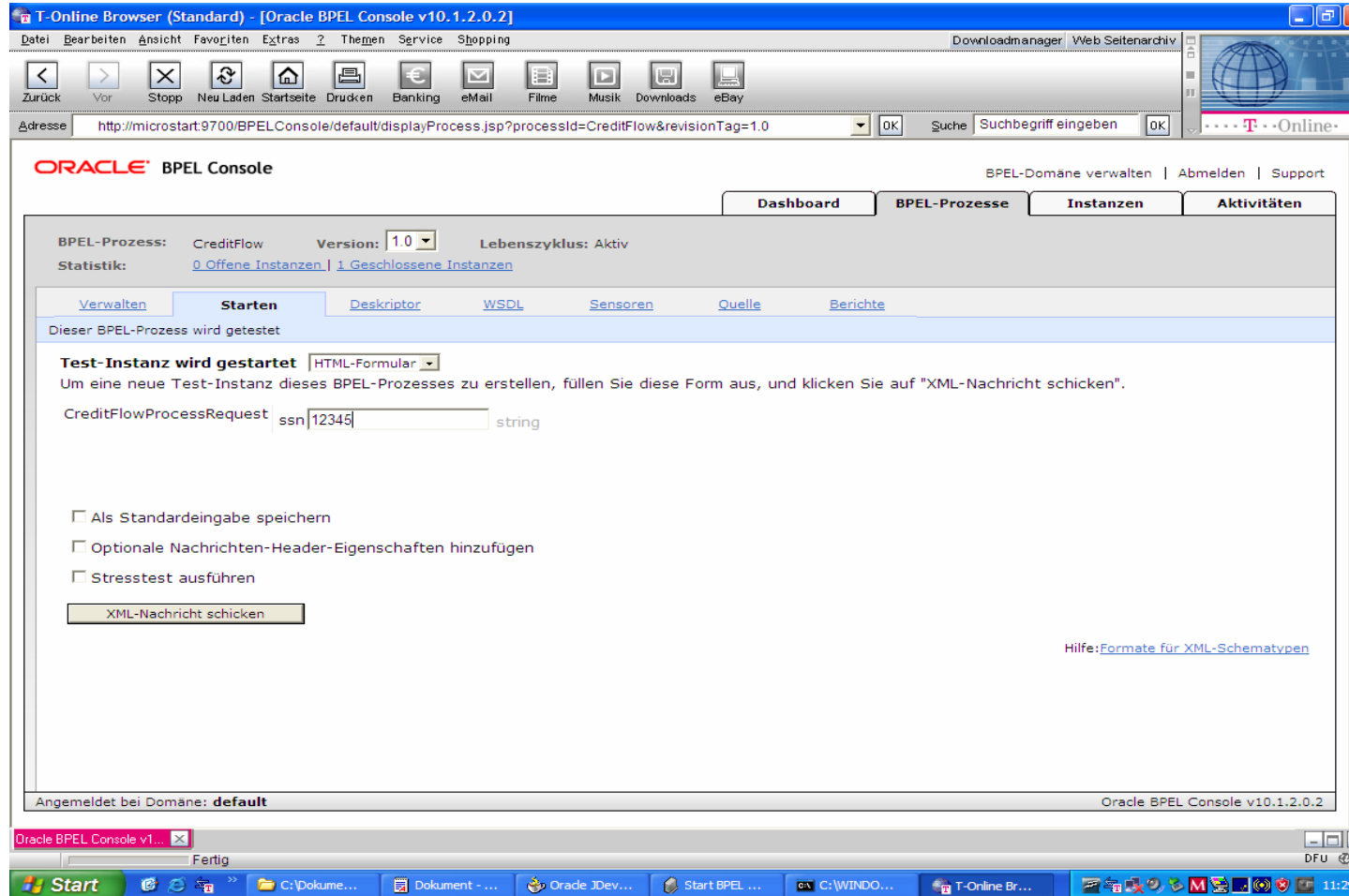
**Laufende BPEL-Prozessinstanzen:**

Instanz	BPEL-Prozess	Zuletzt geändert am ↑
<b>Zuletzt abgeschlossene BPEL-Prozessinstanzen (Mehr...)</b>		
⚠ 603 : Instance #603 of BPELAuftragsabwicklung	BPELAuftragsabwicklung (v. 1.0)	04.12.06 21:31:47
602 : Instance #602 of BPELAuftragsabwicklung	BPELAuftragsabwicklung (v. 1.0)	04.12.06 21:28:05
601 : Instance #601 of BPELAuftragsabwicklung	BPELAuftragsabwicklung (v. 1.0)	04.12.06 21:26:19
503 : Instance #503 of BPELAuftragsabwicklung	BPELAuftragsabwicklung (v. 1.0)	04.12.06 16:57:50
502 : Instance #502 of BPELAuftragsabwicklung	BPELAuftragsabwicklung (v. 1.0)	04.12.06 16:55:20

Angemeldet bei Domäne: **default** Oracle BPEL Console v10.1.2.0.2

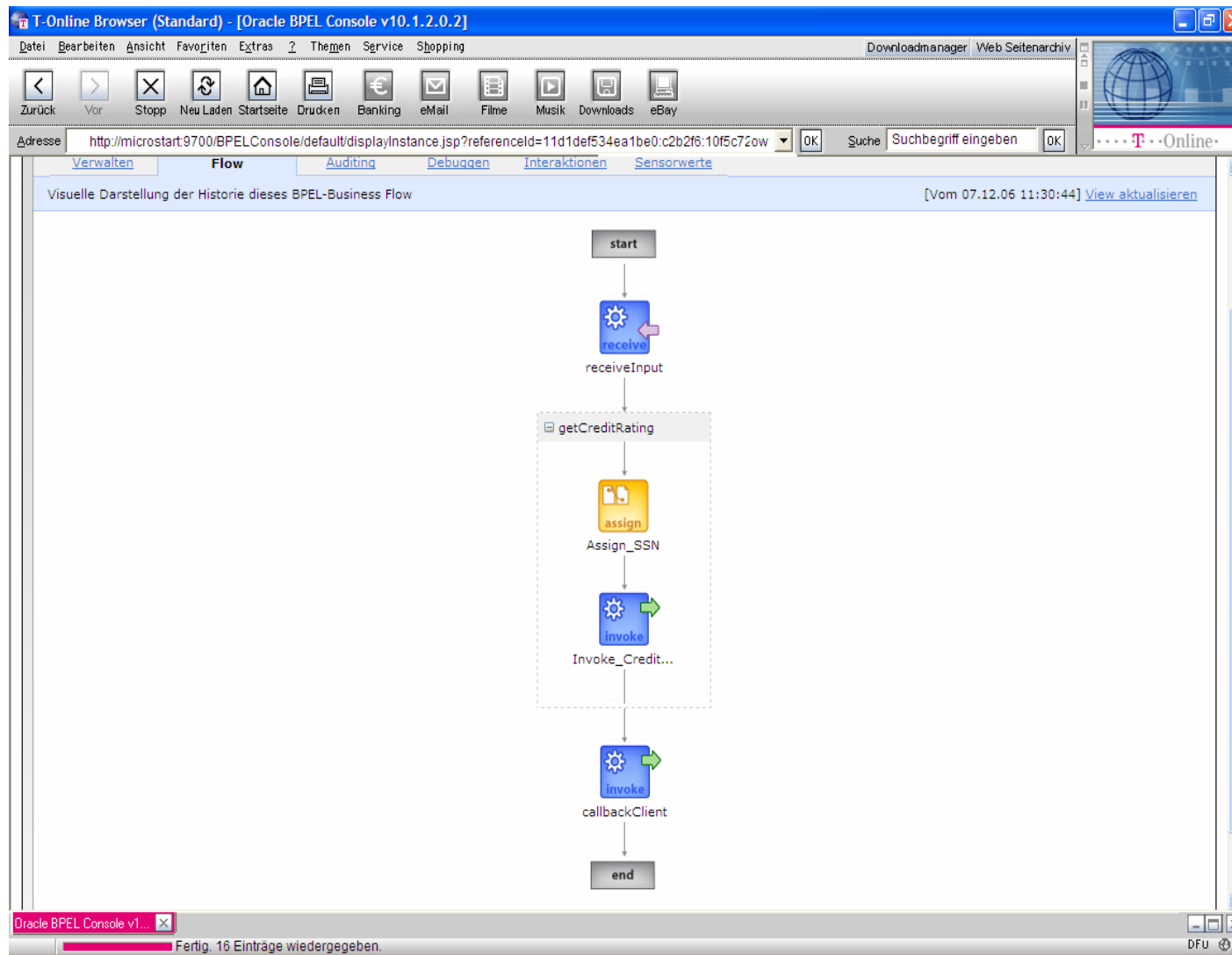
# 8.1 Business Process Execution Language

## Beispiel: CreditFlow (BPEL Console - Testinstanz)



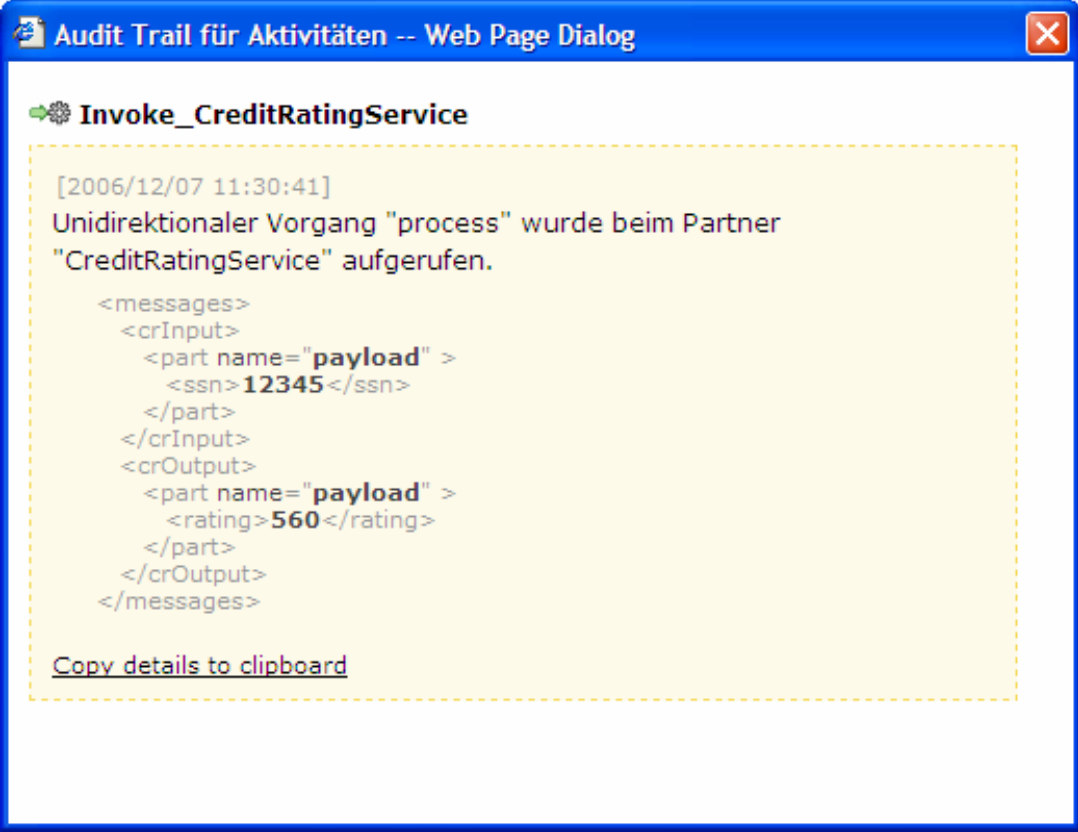
# 8.1 Business Process Execution Language

## Beispiel: CreditFlow (BPEL Console - Audit Trail)



# 8.1 Business Process Execution Language

## Beispiel: CreditFlow (BPEL Console - Audit Trail)



Audit Trail für Aktivitäten -- Web Page Dialog

Invoke\_CreditRatingService

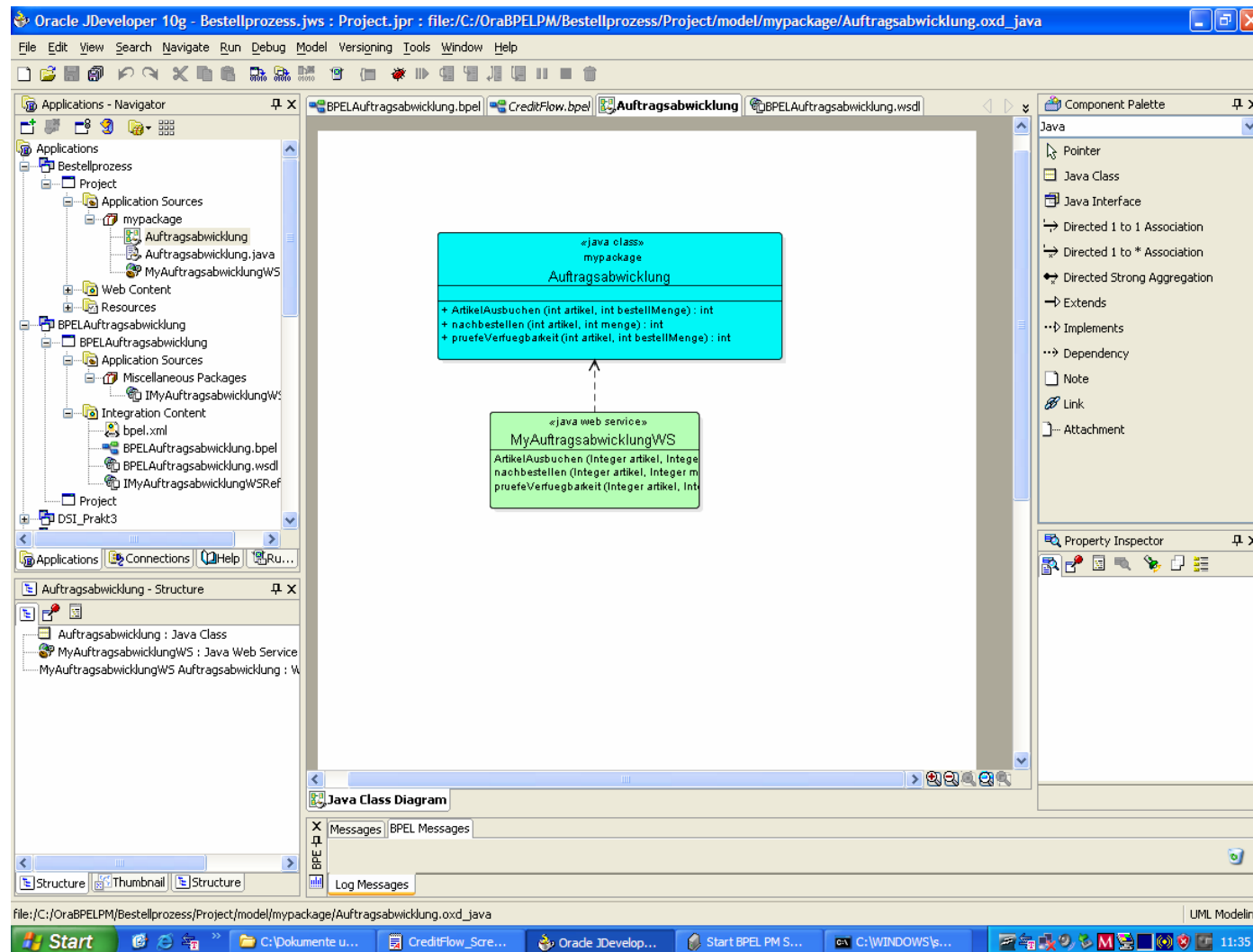
[2006/12/07 11:30:41]  
Unidirektionaler Vorgang "process" wurde beim Partner "CreditRatingService" aufgerufen.

```
<messages>
  <crInput>
    <part name="payload" >
      <ssn>12345</ssn>
    </part>
  </crInput>
  <crOutput>
    <part name="payload" >
      <rating>560</rating>
    </part>
  </crOutput>
</messages>
```

[Copy details to clipboard](#)

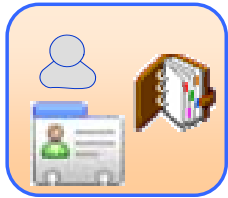
# 8.1 Business Process Execution Language

## Beispiel: JDeveloper (Entwicklung eines Webservice)



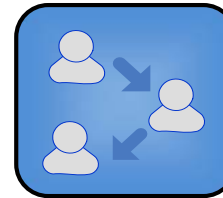
# 8.1 Business Process Execution Language

## Oracle BPM – Workflow Features



### Task Assignment

- Users
- Roles
- Groups



### Task routing

- Declarative patterns
- Ad-hoc routing
- Document based routing
- Dispatching



### Management Rules

- Escalation
- Delegation
- Vacation
- Work load balancing



### Notifications

- Declarative specification of:
  - When – assigned, expired, ...
  - Who – assignee, manager, ...
- Email, Voice, Pager, SMS



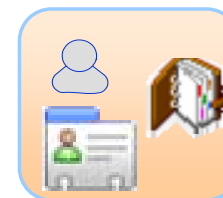
### Work-list Application

- Profile based – assignee, supervisor, group owner, process owner
- Auto-generated JSP forms
- Integration with ADF
- Comments & Attachments
- Available as portlets
- Completely customizable
- Web Services and Java API



### Reports, Audit Trails, ...

- Productivity and distribution reports
- Complete history and audit trail



### Identity Management

- Roles and Org. Hierarchy
- Integrates with OID, LDAP, JAZN; custom plug-ins for other directories

# 8.1 Business Process Execution Language BPEL Test Framework

## 1. Create XML unit test driver from instance audit trail

The screenshot displays the Oracle BPEL Console interface. At the top left is the 'ORACLE BPEL Console' logo. On the top right, there are links for 'Manage BPEL Domain', 'Logout', and 'Support'. Below the logo, there are four tabs: 'Dashboard', 'BPEL Processes', 'Instances', and 'Activities'. The 'Instances' tab is selected, showing details for 'Instance #25223 of UnitedLoan'. The details include: 'Reference Id: 25223' with a 'Tree Finder' icon, 'BPEL Process: UnitedLoan (v. 1.0)', 'Last Modified: 1/5/06 5:58:38 PM', 'State: closed.completed', and 'Priority: 3'. Below the details, there are several sub-tabs: 'Manage', 'Flow', 'Audit', 'Debug', 'Interactions', 'Sensor Values', and 'Test'. The 'Test' tab is selected, showing 'Test Case Information'. The main content area contains a paragraph explaining that the instance can be saved as a test case and imported into a JDeveloper project. It also provides two links: 'Save as unit test (.xml)' and 'Save as integration test (.zip)', each with a brief description of their use.

**ORACLE BPEL Console** Manage BPEL Domain | Logout | Support

**Dashboard** **BPEL Processes** **Instances** **Activities**

**Title:** Instance #25223 of UnitedLoan **Last Modified:** 1/5/06 5:58:38 PM  
**Reference Id:** 25223 [Tree Finder](#) **State:** closed.completed  
**BPEL Process:** [UnitedLoan \(v. 1.0\)](#) **Priority:** 3 [more](#)

[Manage](#) [Flow](#) [Audit](#) [Debug](#) [Interactions](#) [Sensor Values](#) **Test**

Test Case Information

This instance may be saved as a test case. After downloading, you can import it into the process's JDeveloper project and add assertions to complete the test case's implementation.

[Save as unit test \(.xml\)](#) Saving a test as a unit test will create a single test case for this process that emulates all partner interactions. This type of test can be useful when testing the process by itself.

[Save as integration test \(.zip\)](#) Saving a test as an integration test will create multiple test cases, one for this instance and one for each partner that is a BPEL process. This type of test can be useful when testing an entire system.

# 8.1 Business Process Execution Language BPEL Test Framework

## 2. Execute tests

**ORACLE** BPEL Console Manage BPEL Domain |

**Dashboard** | **BPEL Processes** | **Instances**

**BPEL Process:** BuggyLoanFlow    **Version:** 1.0    **Lifecycle:** Active  
**Statistics:** [1 Open Instances](#) | [5 Closed Instances](#)

[Manage](#)   [Initiate](#)   [Descriptor](#)   [WSDL](#)   [Sensors](#)   [Source](#)   **Test Suites**

Test Suites

**Execute Tests**   **Undeploy Tests**   maximum concurrent instances:

[expand all](#)   [collapse all](#)   [select all](#)   [deselect all](#)

- ExceptionFlows**  
This suite tests exception handling. It is critical these tests pass.  

Test Name	Test Description
<input type="checkbox"/> testCreditRatingFaultHandler.xml	[no description]
- MainFlows**  
This suite tests the loan selection logic. It is critical these tests pass.  

Test Name	Test Description
<input checked="" type="checkbox"/> testCreditRatingServiceInput.xml	[no description]
<input checked="" type="checkbox"/> testLoanServicesInput.xml	[no description]
<input checked="" type="checkbox"/> testStarLoanOfferSelected.xml	[no description]

# 8.1 Business Process Execution Language BPEL Test Framework

## 3. View results, code coverage, fix as needed and repeat

### BPEL Test Report

#### Test Parameters

Process ID: BuggyLoanFlow (v. 1.0)  
Start date: Thursday, January 05, 2006 6:00:43 PM  
Completion Date: Thursday, January 05, 2006 6:00:56 PM  
Test Suites: MainFlows  
Workers: 1

#### Summary Report

Total	Initiated	Completed	Successful	Success Rate	Code Coverage
5	5	5	3	60%	<a href="#">92%</a>

#### Detail Report

Display Failures Only

✓ testCreditRatingServiceInput (MainFlows)	<a href="#">View Instance Flow</a>
✓ testLoanServicesInput (MainFlows)	<a href="#">View Instance Flow</a>
✓ testStarLoanOfferSelected (MainFlows)	<a href="#">View Instance Flow</a>
✗ testUnitedLoanOfferSelected (MainFlows) <a href="#">Less</a>	<a href="#">View Instance Flow</a>

Message	Expected	Actual
The APR is not correct!	5.70	55.7
The Loan Provider is not correct! United Loan Star Loan		



# Vorlesung „ Daten- und Systemintegration“

## Kapitel 9 Business Prozess Management

- 8.1 BPEL (Business Process Execution Language)
- 8.2 BPEL Server-Produkte**
- 8.3 BPEL4PEOPLE
- 8.4 BPELJ
- 8.5 Transformation eEPK->BPEL, BPMN->BPEL
- 8.6 Links+Artikel

## 8.2 BPEL Server-Produkte

### **BPEL Servers (Beispiele)**

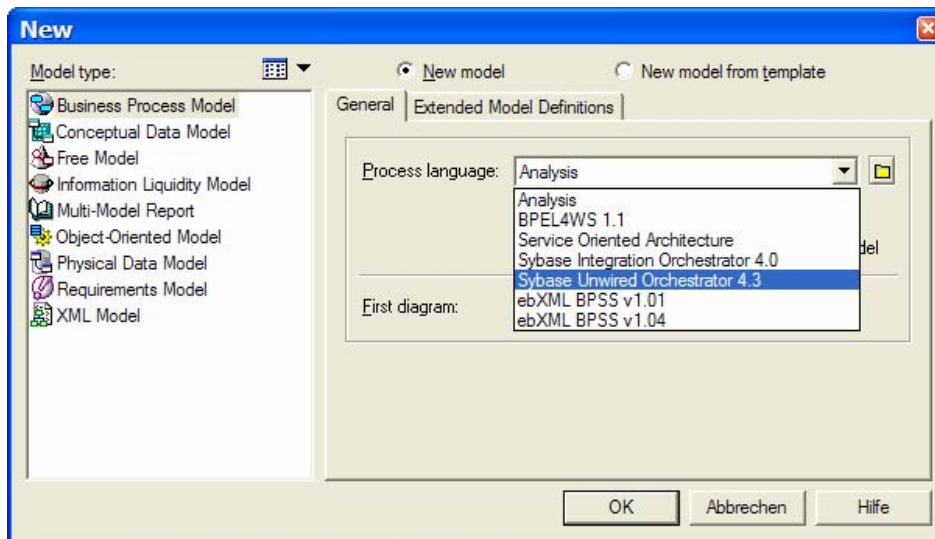
- ActiveBPEL Engine
- IBM WS-BPEL Editor and Java Run-Time
- Oracle BPEL Process Manager
- Microsoft BizTalk Server
- JBoss jBPM

## 8.2 BPEL Server-Produkte

### Sybase PowerDesigner (PD11 – BPM)

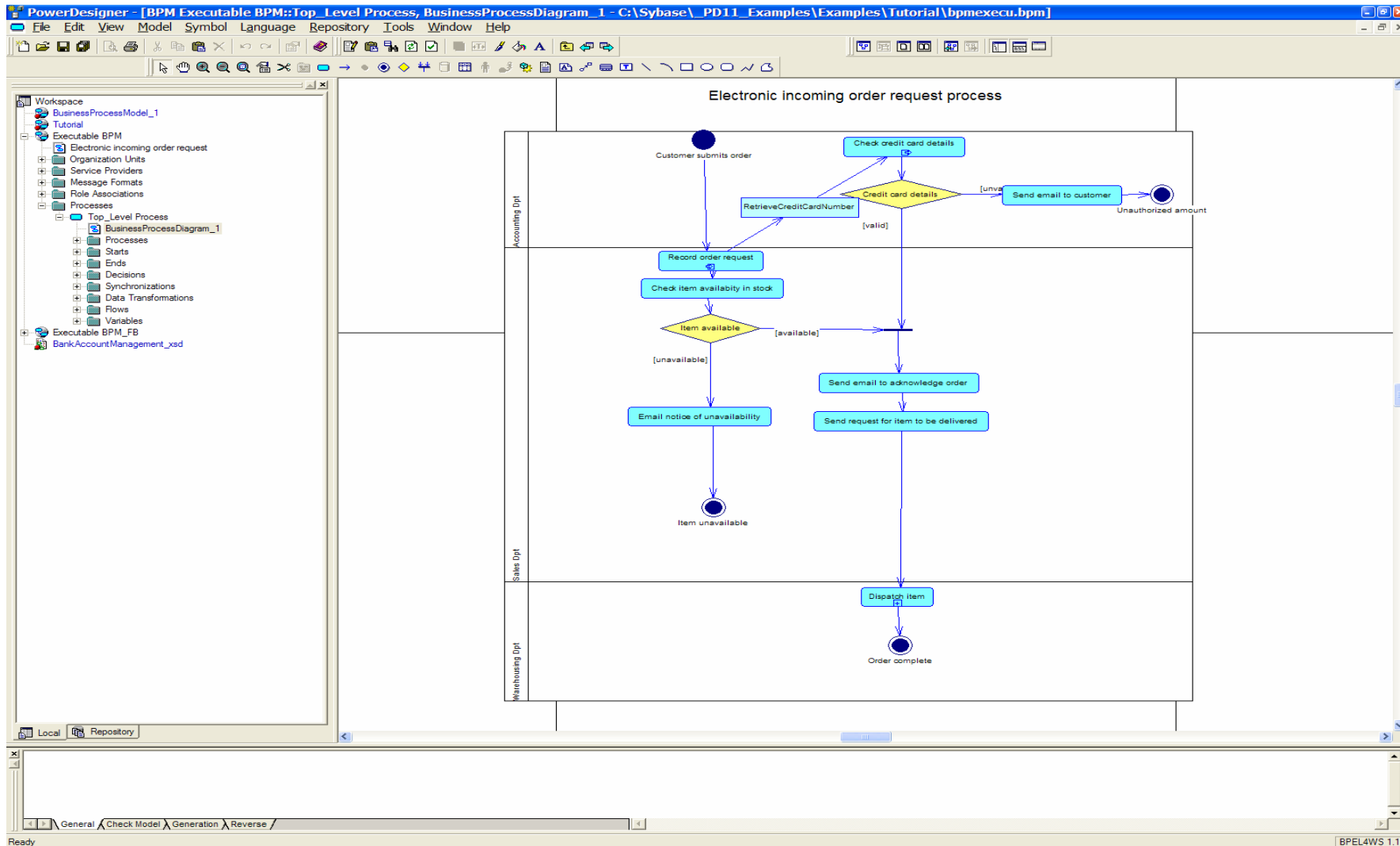
Der PowerDesigner unterstützt

- die Modellierung von Geschäftsprozessen auf Basis der UML („konzeptionelles GPM“)
- die Spezifikation von ausführbaren Prozessen auf Basis der WS-BPEL bzw. einer speziellen “Unwired Orchestrator Syntax” („ausführbares GPM“).



# 8.2 BPEL Server-Produkte

## Sybase PowerDesigner (PD11 – BPM)



## 8.2 BPEL Server-Produkte

### Unwired Orchestrator Overview

“Unwired Orchestrator provides integration capabilities, **Orchestration**, and **Business Activity Monitoring** (BAM) in an easy-to-use and unified tool.

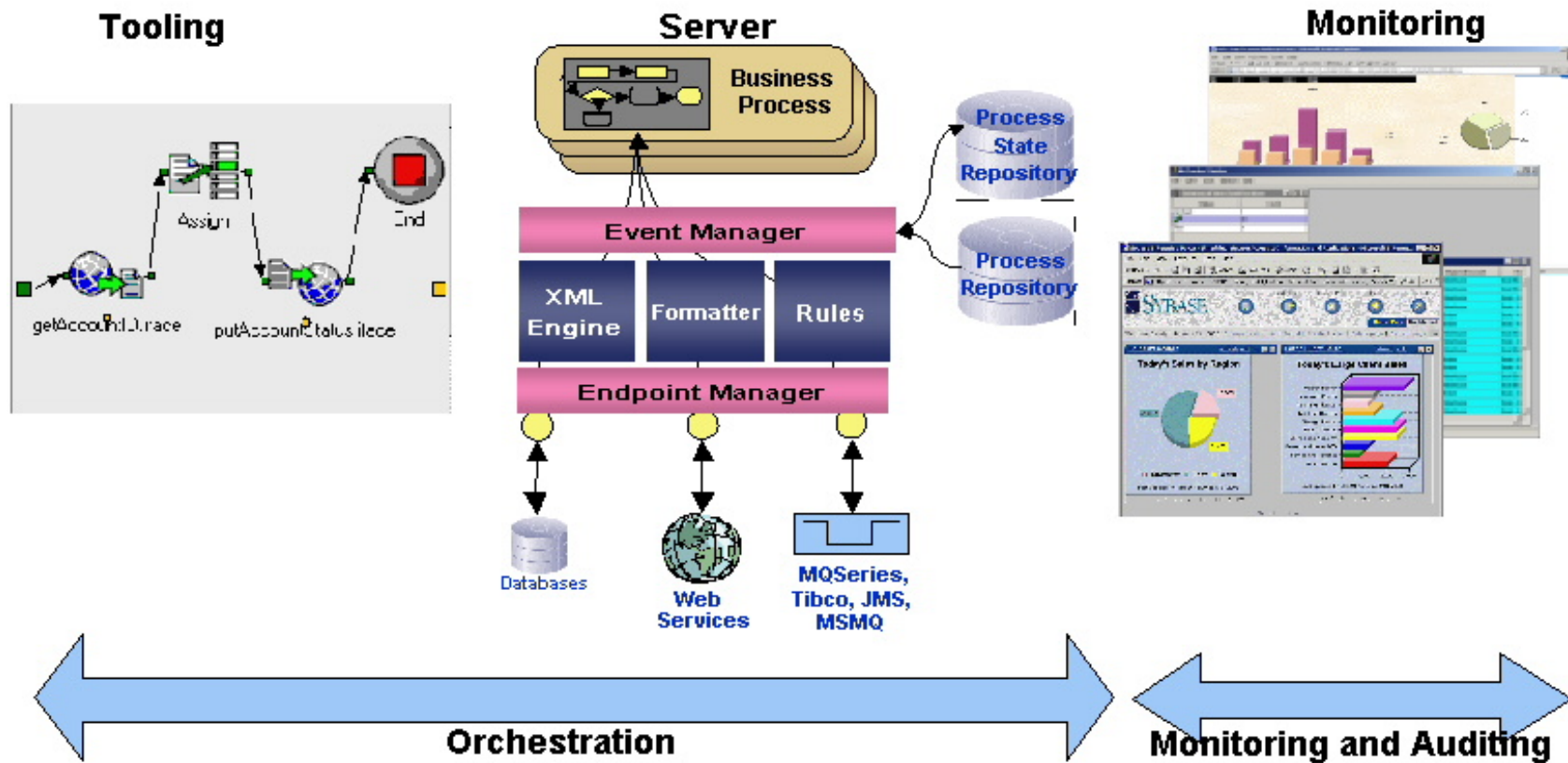
Its open architecture enables integration of both internal and external systems into a single operational unit, which results in organizational efficiency and reduced costs.

The user interfaces provide the ability to **manage the life cycle of a business process from design, deployment, management, and monitoring**, to refinement in a logical manner.

The expanded Monitoring and Auditing capabilities allow data to be analyzed not only at the message level but also at the process level. Additionally, the user interface provides a simplified way to extract and display information for performing real-time business analysis.”

# 8.2 BPEL Server-Produkte

## Unwired Orchestrator Overview Diagram



# 8.2 BPEL Server-Produkte

## Sybase Unwired Orchestrator

The screenshot displays the Sybase Unwired Orchestrator interface. The main workspace shows a BPEL process diagram with three components: an input service interface (INService.iface), an Assign activity, and an output service interface (OUTService.ifa). The diagram is connected to a Properties window on the right, which lists various properties for the selected component.

INService.iface Property	Value
Initializer	Yes
Interface Type	Not
Root Tag	dbS
Target Context Reference	/IN
Target Context Reference Header	
Ancillary Activities	
Print to console	
Write to file	
Write to log	
Write to probe	
Diagram	
Description	
Icon	
Label	
Use Default Icon	INS
Read-only	Yes
Id	IFa
Type	Inte

At the bottom of the interface, there is a table for tasks:

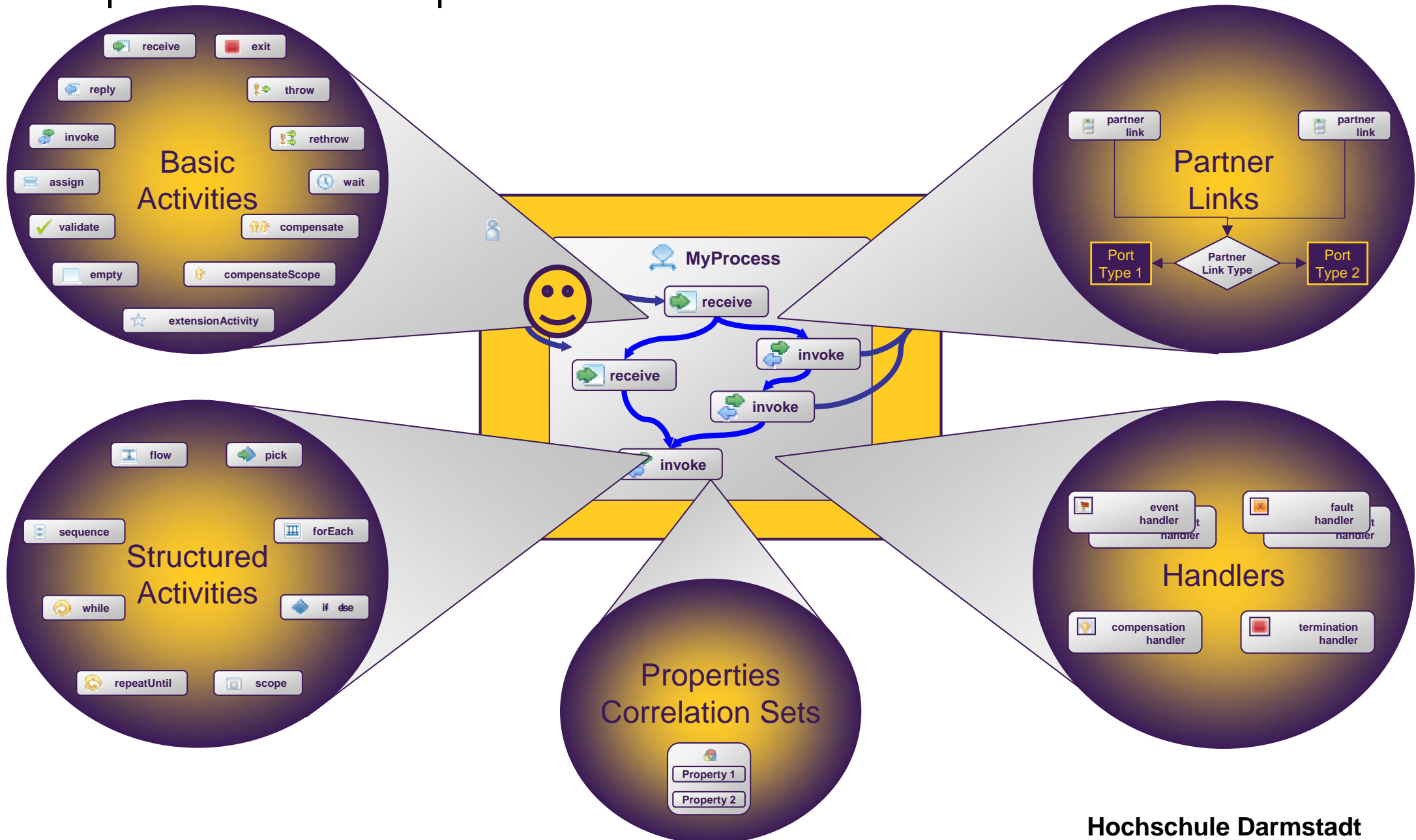
Tasks (0 items)	Description	Resource	In Folder	Location

# Vorlesung „ Daten- und Systemintegration“

## Kapitel 9 Business Prozess Management

- 8.1 BPEL (Business Process Execution Language)
- 8.2 BPEL Server-Produkte
- 8.3 BPEL4PEOPLE**
- 8.4 BPELJ
- 8.5 Transformation eEPK->BPEL, BPMN->BPEL
- 8.6 Links+Artikel

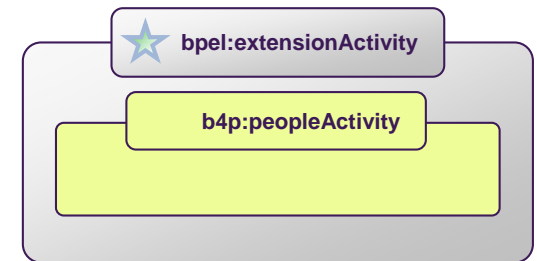
# Kap. 8.3: BPEL4People & WS-HumanTask



# Kap. 8.3: BPEL4People & WS-HumanTask

## Motivation

- Integration von **menschlichen Interaktionen** für Web-basierte Geschäftsprozesse sowie SOA-Anwendungen
- **Standard-Lösung** für verschiedenste Szenarien, bei denen Portabilität wichtig ist



## BPEL4People

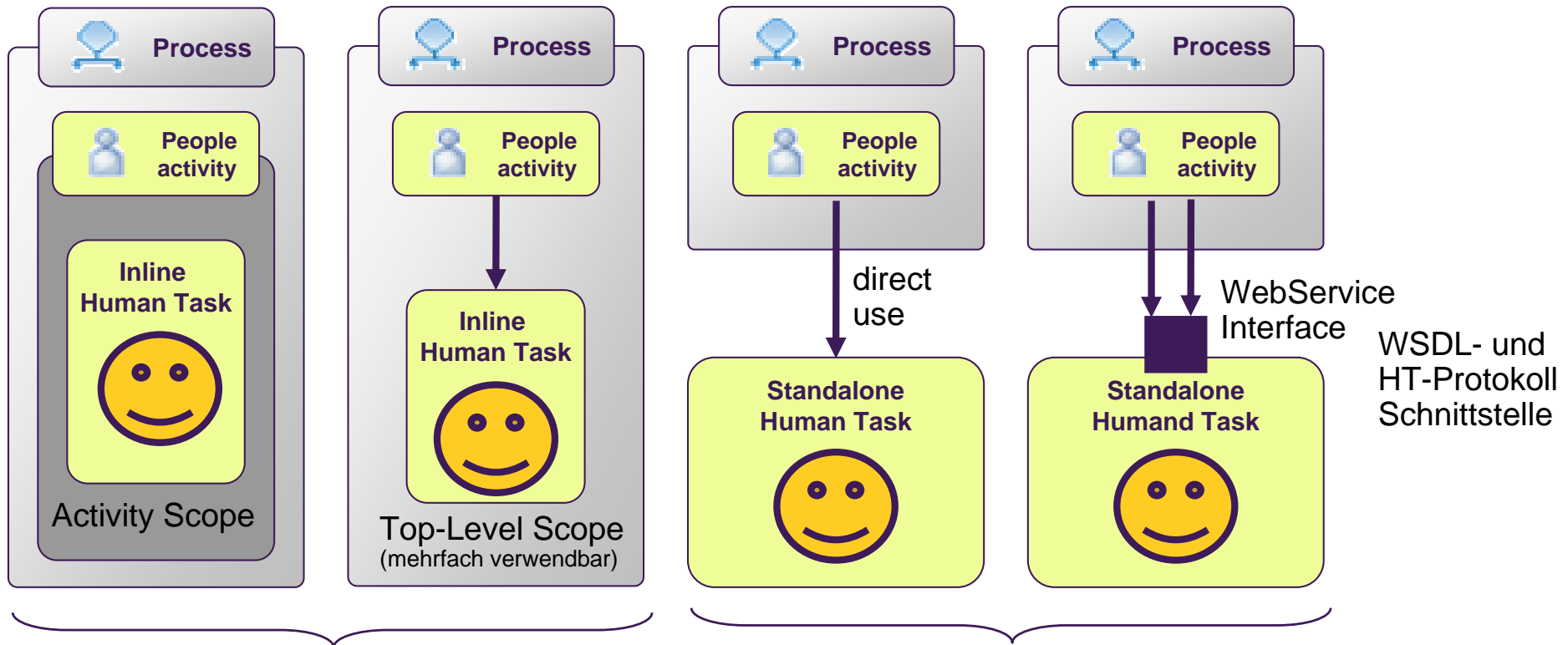
Definition von menschlichen Aktionen innerhalb von BPEL-Prozessen

Integration **auf Basis der WS-BPEL 2.0-Spezifikation**

## WS-HumanTask

- Definition von Service-basierten Aufgaben und Benachrichtigungen sowie eines Protokolls zur Steuerung der Human-Task
- Bereitstellung einer Schnittstelle für Client-Anwendungen (z. B. Worklist-Anwendungen)

# Kap. 8.3: BPEL4People & WS-HumanTask



Teil eines BPEL-Prozesses  
mittels PEOPLE Activity

Human Task außerhalb eines BPEL-  
Prozesses (kein Zugriff auf BPEL-  
Prozesskontext)



# Vorlesung „ Daten- und Systemintegration“

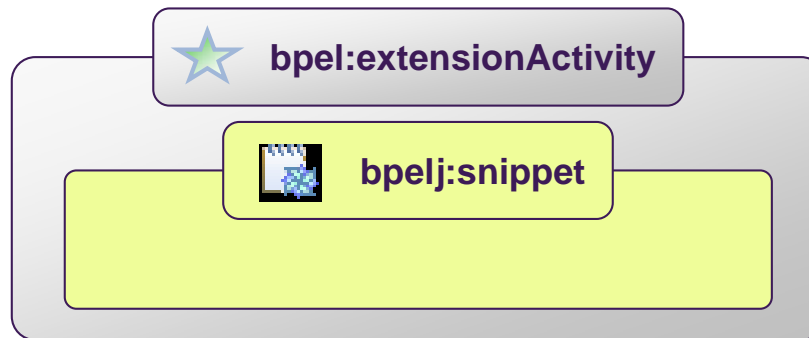
## Kapitel 9 Business Prozess Management

- 8.1 BPEL (Business Process Execution Language)
- 8.2 BPEL Server-Produkte
- 8.3 BPEL4PEOPLE
- 8.4 BPELJ**
- 8.5 Transformation eEPK->BPEL, BPMN->BPEL
- 8.6 Links+Artikel

# Kap. 8.4: BPELJ

## Motivation

Verwendung von Java-Inline-Code in BPEL-Prozessen, um fortgeschrittene Konzepte (z. B. Transaktionskonzepte) direkt zu unterstützen.



WS-BPEL 2.0 Extensions for Java (BPELJ)

<http://www-128.ibm.com/developerworks/library/specification/ws-bpelj>

# Kap. 8.4: BPELJ

## BPELJ Examples – Inline Java Expressions

### Beispiel: Condition (“Boolean Expression”)

```
<if>
  <condition
    expressionLanguage="urn:oasis:names:tc:wsbpel:2.0:sublang:java1.4">
    widget.equals(getVariableProperty("PO", "productName"))
  </condition>
  ...
<else>...</else>
</if>
```

### Beispiel: Variable initialization (“General Expression”)

```
<variable name="salesTax" type="xsd:float">
  <from
    expressionLanguage="urn:oasis:names:tc:wsbpel:2.0:sublang:java1.4">
    subtotal * taxRate
  </from>
</variable>
```

# Kap. 8.4: BPELJ

## BPELJ Examples – Inline Java Activity

### Beispiel: „Extension Activity“

```
<process name="purchaseOrderProcess" bpelj:xmlBinding="bpelj:DOM3" ...>
  <variables>
    <variable name="justificationDoc" type="lns:justificationDocument"/>
    <variable name="po" type="lns:POMsg" bpelj:xmlBinding="bpelj:SDO1.0"/>
  </variables>
  <sequence>
    ...
    <extensionActivity>

      <bpelj:snippet>
        // Get the approver using SDO accessor
        Approver approver = po.getApprover();
        // Get the approver's comments
        NodeList commentNodeList =
          justificationDoc.getElementsByTagName("approverComment");
        ...
      </bpelj:snippet>

    </extensionActivity>
  </sequence>
</process>
```



# Vorlesung „ Daten- und Systemintegration“

## **Kapitel 9      Business Prozess Management**

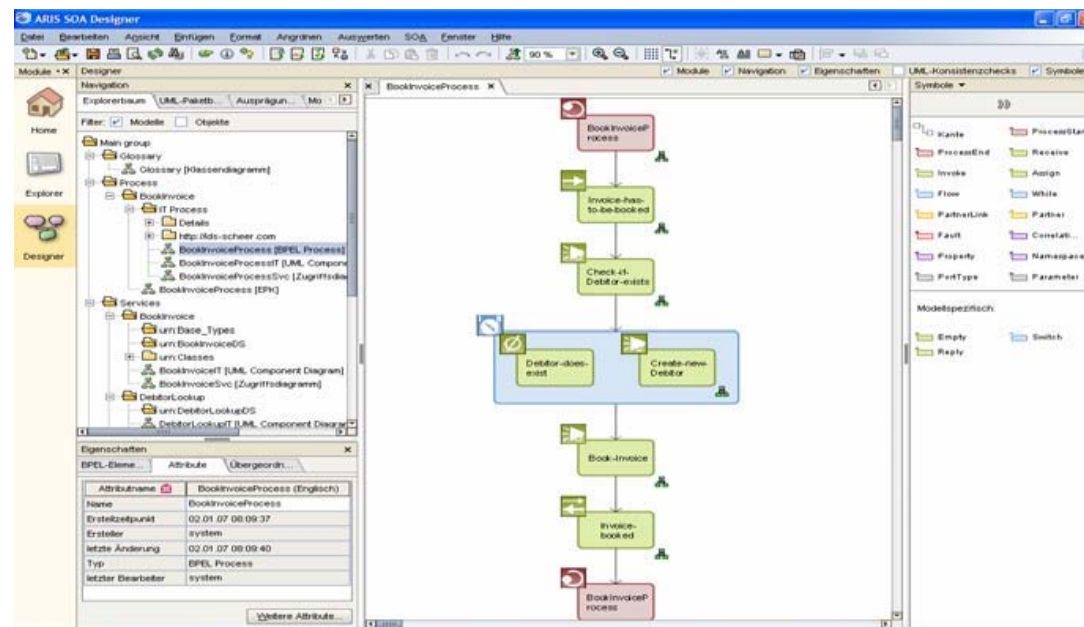
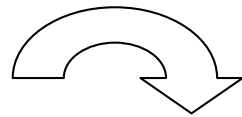
- 8.1      BPEL (Business Process Execution Language)
- 8.2      BPEL Server-Produkte
- 8.3      BPEL4PEOPLE
- 8.4      BPELJ
- 8.5      Transformation eEPK->BPEL, BPMN->BPEL**
- 8.6      Links+Artikel

# Kap. 8.5: EPK->BPEL

## Übergang EPK->BPEL

Der SOA-Designer ermöglicht den halb-automatisierten Übergang von EPK-Modellen nach BPEL (s. a. Dokument zum Download).

EPK-Modell

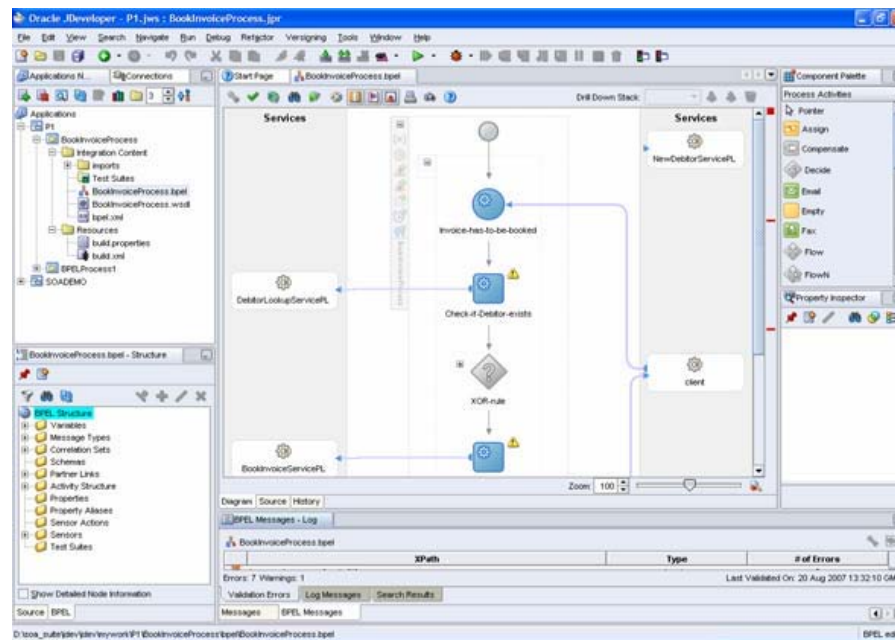
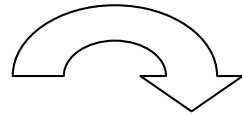


# Kap. 8.5: EPK->BPEL

## Übergang EPK->BPEL

Nach dem Export des Modells nach BPEL, kann das Ergebnis in den Oracle JDeveloper importiert werden.

BPEL-Modell



# Kap. 8.5: EPK->BPEL

## Übergang EPK->BPEL

### Bewertung

- Der SOA-Designer bietet die Möglichkeit der Transformation von EPK-Modellen nach BPEL.
- Allerdings erscheint der Aufwand für die zusätzlich notwendige Spezifikationen für das Mapping recht hoch zu sein.
- Negativ ist, dass der Anwender relativ viel BPEL-KnowHow besitzen muss.
- Der Import in die Oracle SOA Suite hat problemlos funktioniert.

# Vorlesung „ Daten- und Systemintegration“

## **Kapitel 9      Business Prozess Management**

- 8.1      BPEL (Business Process Execution Language)
- 8.2      BPEL Server-Produkte
- 8.3      BPEL4PEOPLE
- 8.4      BPELJ
- 8.5      Transformation eEPK->BPEL, BPMN->BPEL
- 8.6      Links+Artikel**

## 8.6 Links

### Links

IBM, SOA and Web Services - <http://www-306.ibm.com/software/solutions/webservices/>

*Business Process Execution Language (BPEL4WS 1.1) –*  
<http://www-128.ibm.com/developerworks/library/specification/ws-bpel/>

*The Workflow Reference Model + XPD L -* <http://www.wfmc.org>

*Business Process Model Language Specification -* <http://www.bpmi.org/>

*BPEL learning guide,*  
[http://searchwebservices.techtarget.com/originalContent/0,289142,sid26\\_gci880731,00.html](http://searchwebservices.techtarget.com/originalContent/0,289142,sid26_gci880731,00.html)