The background of the slide features a faded UML class diagram. It shows several classes: 'Buchungskonto' with attributes like 'IN\_BUCHUNGSTILBEIT', 'KontoID', 'Buchungstag', 'Aktiv', and 'Umschreibung'; 'Fernweidekonto' with attributes 'T\_Buchungskonto', 'Telefonnummer', and 'Aktiv', and methods like 'getKonnector()', 'abrechnen(Ausgangsrechnung, Teilabrechnung)', 'getAbrechnendeGespraech()', and 'getAbrechnendeGespraechSubschritt()'; and 'Telefonat' with attributes 'TelefonID' and 'TelefonNummer'. Arrows indicate associations between these classes.

**camunda**  
The Business Process Company

**E6**  
**Business Process Management mit**  
**JBoss jBPM**

Java Forum Stuttgart  
05.07.2007  
Bernd Rücker

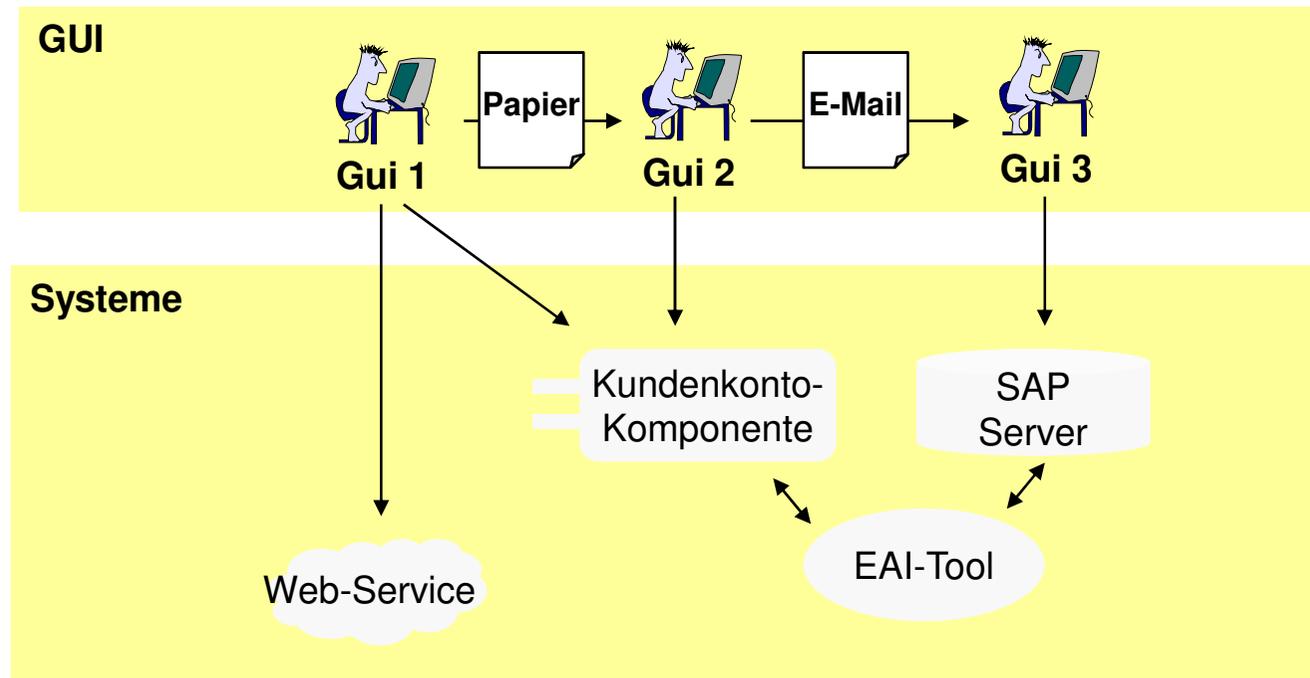
- Geschäftsprozesse (mit Java)
- JBoss jBPM
- Ein Blick in die Java Process Definition Language (jPDL)
- Verwendung von jBPM / Architektur

- camunda GmbH, Bad Mergentheim
- Hauptfokus Beratung, Training & Projektarbeit im Bereich
  - Unternehmensanwendungen mit Java EE
  - business process engines & business rules
- Sehr gute Erfahrung mit
  - JBoss AS
  - JBoss jBPM
  - JBoss Rules (aka Drools)
  - EJB 3, Hibernate, JDO
  - Jack-Rabbit (Java Content Repository / JCR)
  - Open Source Business Intelligence (Pentaho, ...)
- Ich comitte im JBoss jBPM Projekt
- SENS-Experte
- Kontakt:  
bernd.ruecker@camunda.com

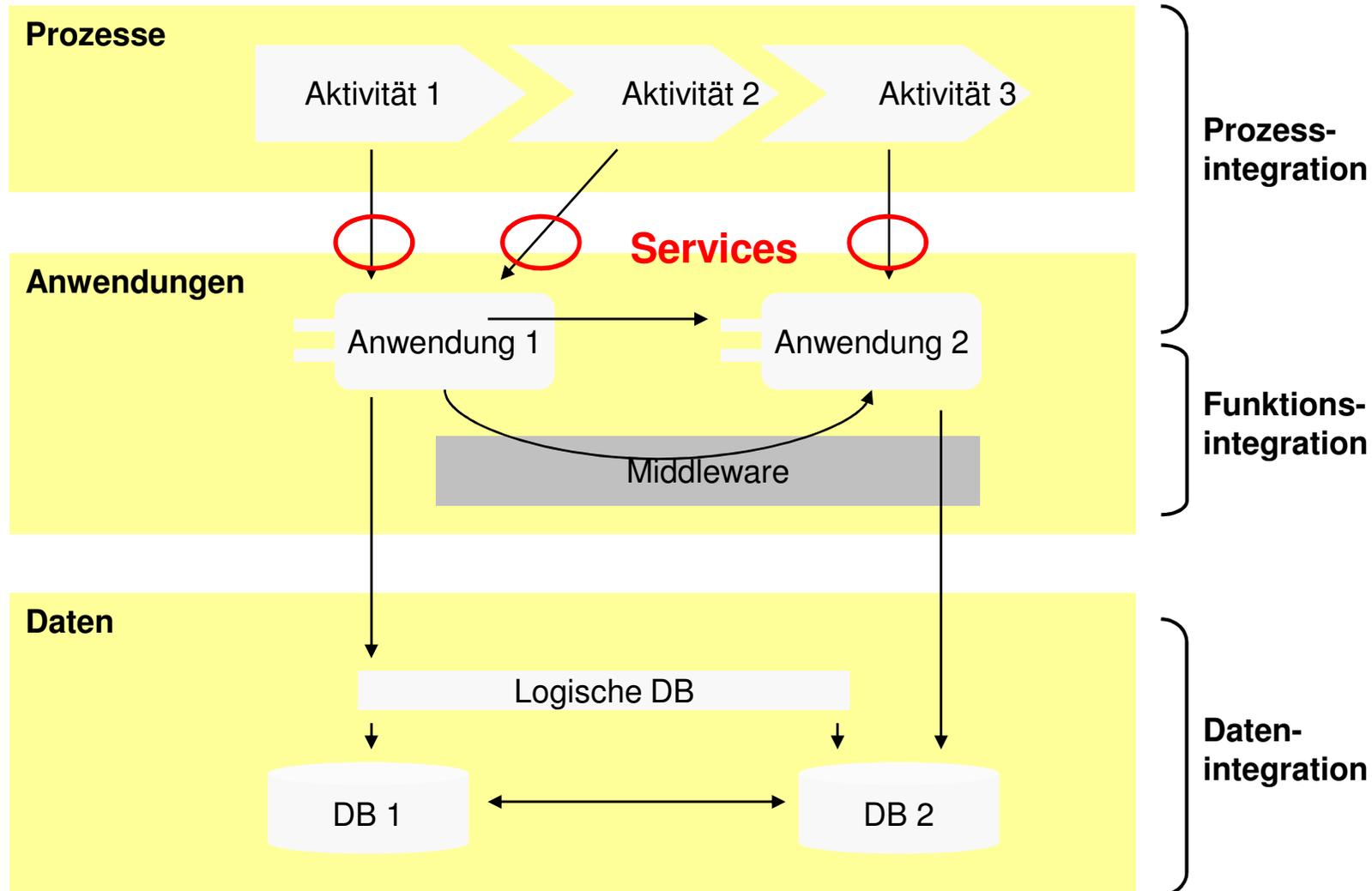


- Aus Input wird durch eine Reihe von Aktivitäten ein definiertes Arbeitsergebnis
- Wertschöpfend
- Lang laufend (bis zu mehreren Monaten)
- Fachlich motiviert
- Aktivitäten werden teilweise durch Menschen, teilweise automatisch durchgeführt

# Status Quo (häufig)



# Integrationsarten



## Java SE

- Keine langlebigen Prozesse
- Persistenz

## Java EE (z.B. Statefull Session Beans)

- Versionierung von Prozessen
- Prozess-Log
- Grafische Prozessbeschreibung
- ...

→ Eigene Business Process Engine

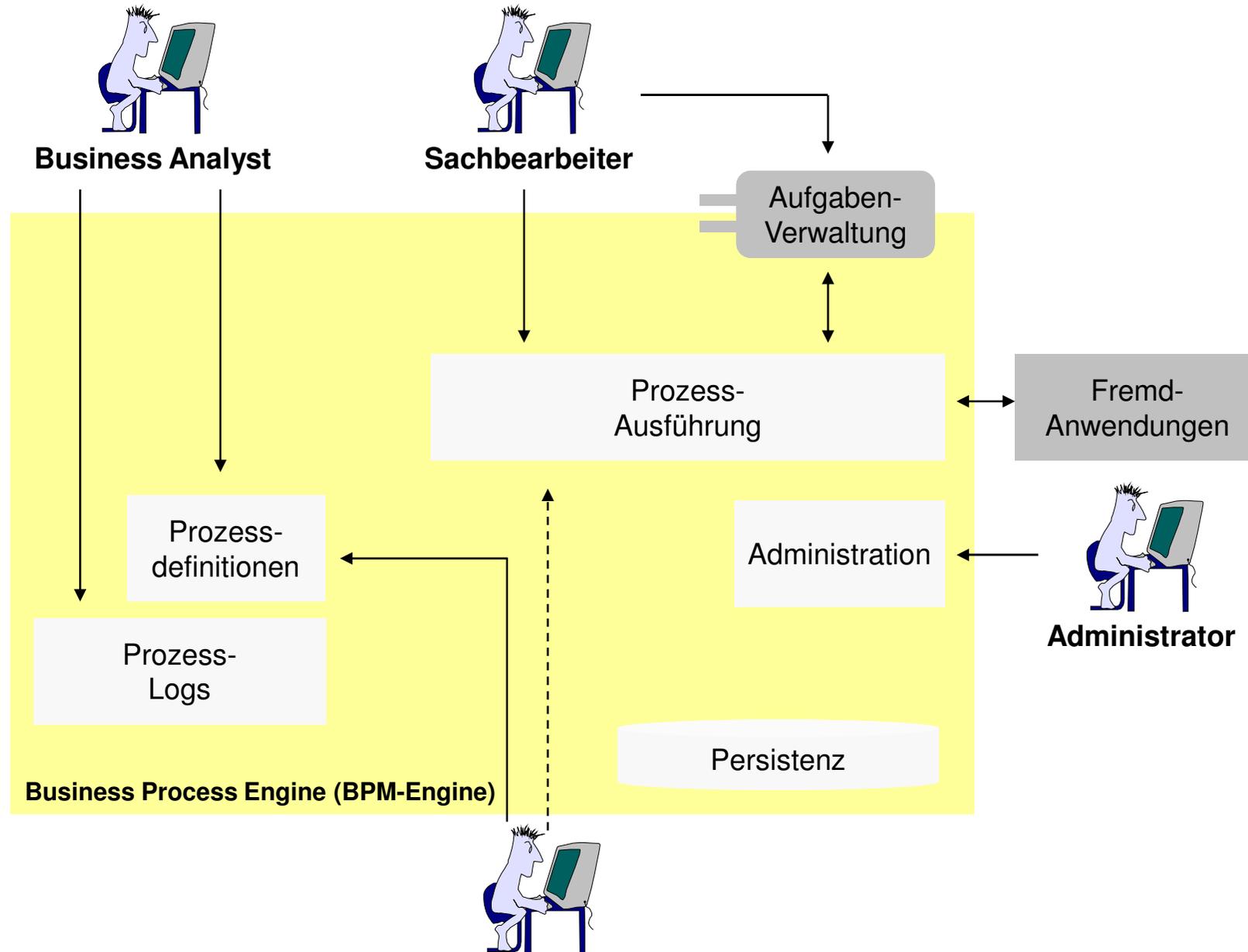
# Was macht eine BP-Engine?

---

- Versionierung, Persistenz & Interpretation von Prozessmodellen
- Steuerung & Persistenz von Prozessinstanzen
- Einbindung externer Services
- Verwalten von Ereignissen (wie Timeouts, ...)
- Prozesskontext (Variablen zu Prozess speichern)
- Bereitstellung eines Logs
- Task-Management

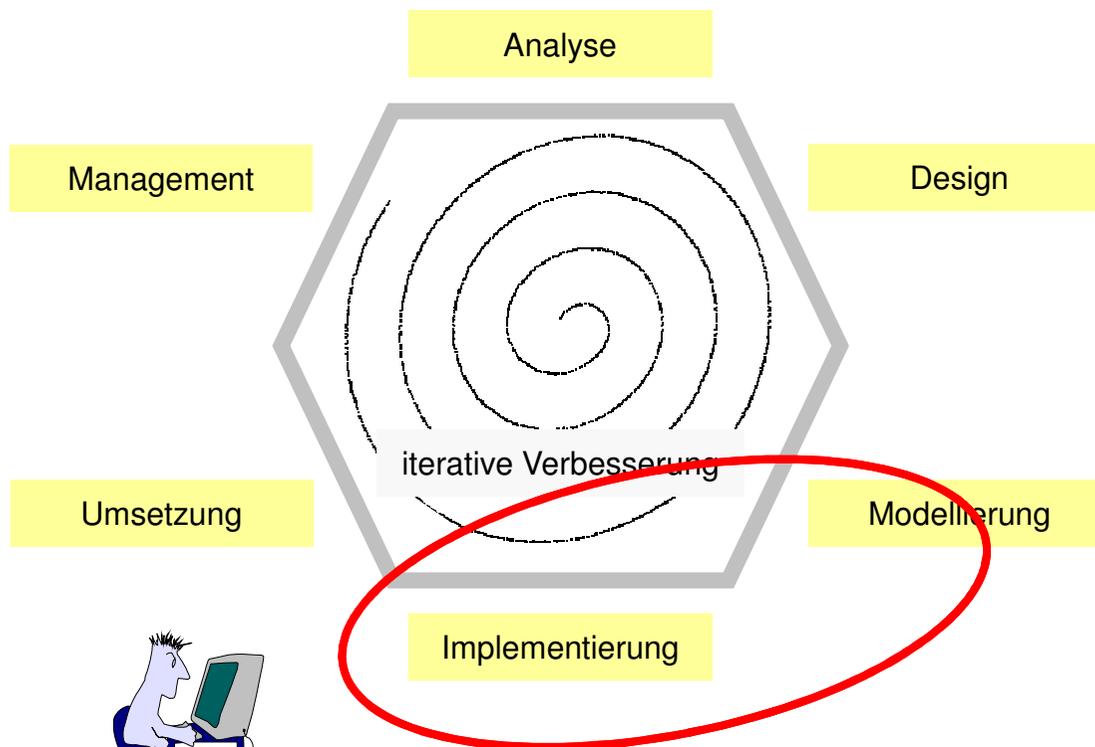
**→ Container für Geschäftsprozesse**

# Business Process Engine





**Business Analyst**



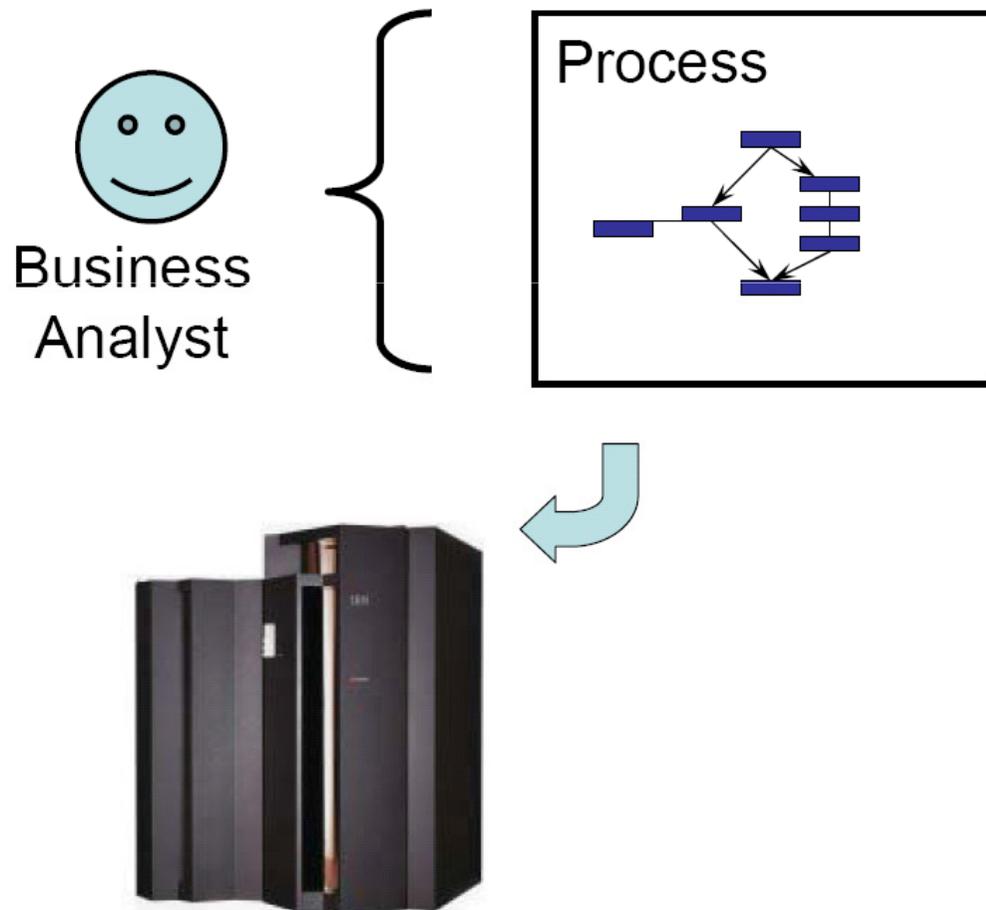
**Developer**

**Business Process Engine**

## Weitere Dienste der BPM-Suiten

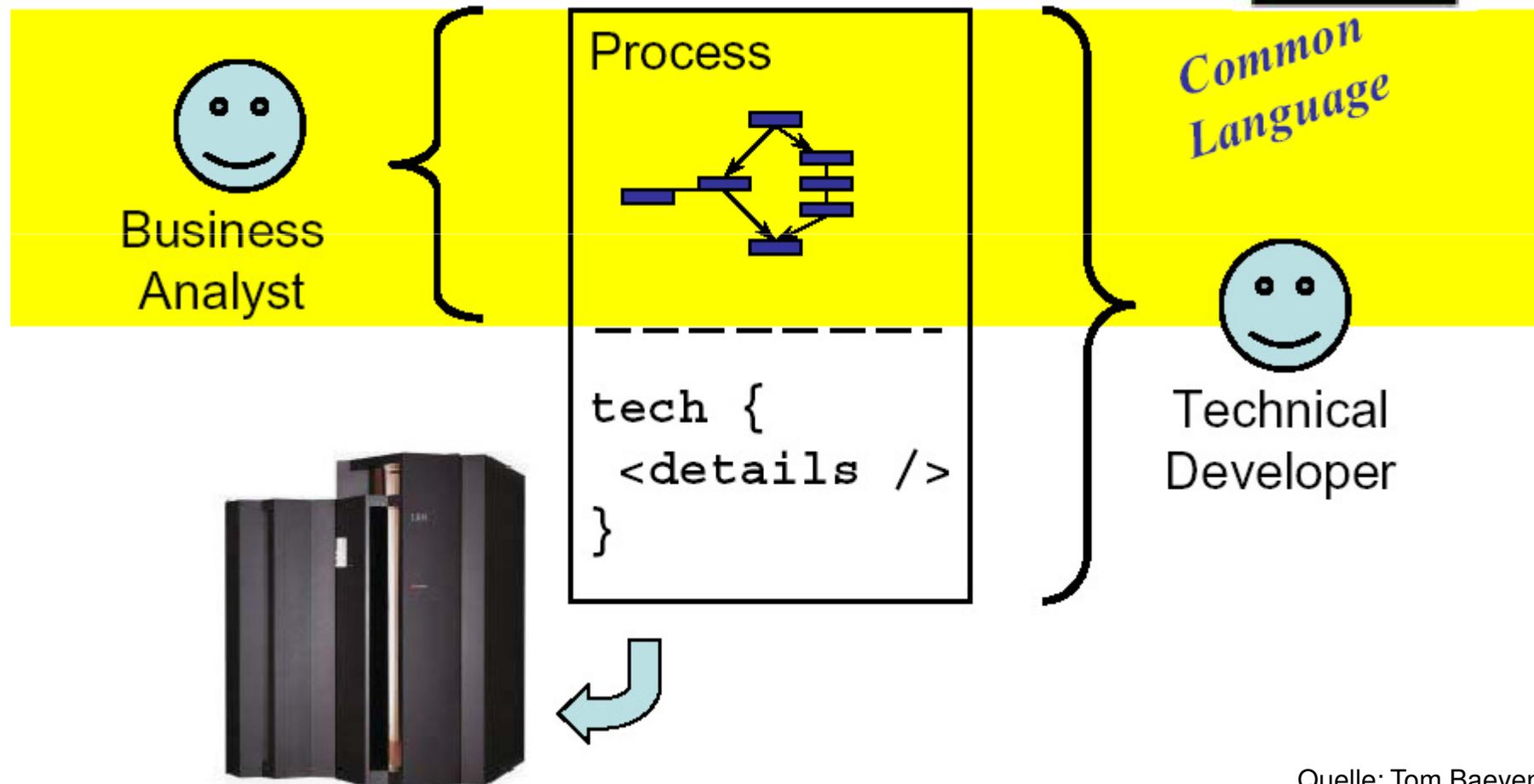
- Business Activity Monitoring (BAM)
- Simulation
- Modellierung

## Collaboration Traditional View

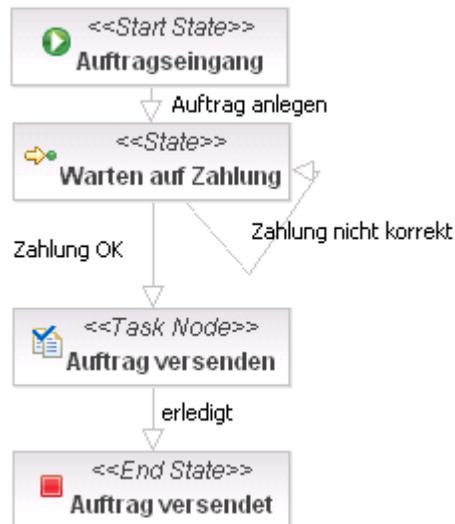


Quelle: Tom Baeyens

## Collaboration Common Language

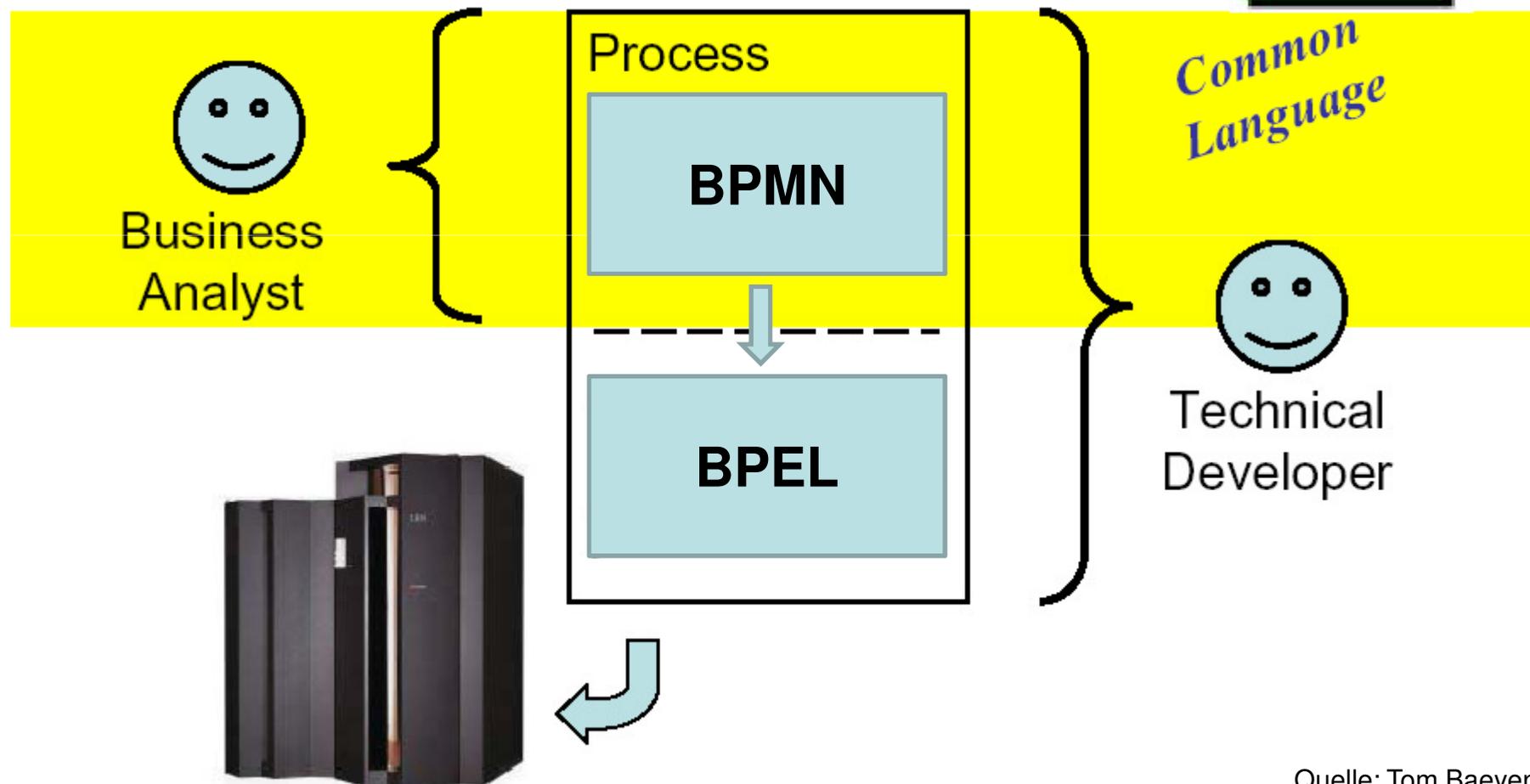


# Konkret: jPDL

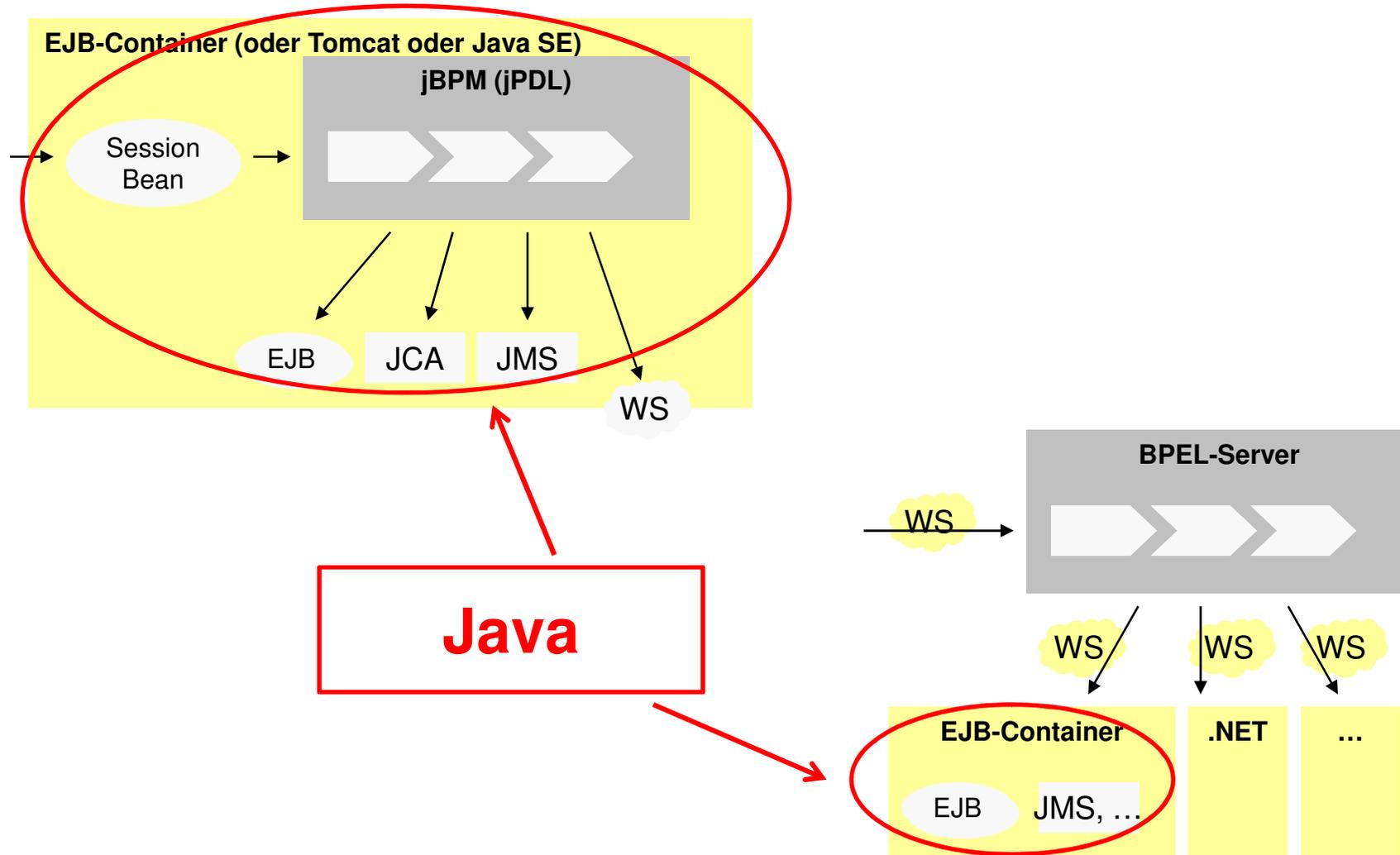


```
<process-definition name="SimpleOrder">
  <swimlane name="Lager">
    <assignment pooled-actors="Lager" />
  </swimlane>
  <start-state name="Auftragseingang">
    <event type="node-leave">
      <action name="create order in business logic" class="com.camunda.jmgui.actions
    </event>
    <transition name="Auftrag anlegen" to="Warten auf Zahlung"></transition>
  </start-state>
  <state name="Warten auf Zahlung">
    <transition name="Zahlung OK" to="Auftrag versenden"></transition>
    <transition name="Zahlung nicht korrekt" to="Warten auf Zahlung"></transition>
  </state>
  <task-node name="Auftrag versenden">
    <task name="Auftrag versenden" swimlane="Lager" />
    <event type="node-leave">
      <action name="set order shipped status in business logic" class="com.camunda.
    </event>
    <transition name="erledigt" to="Auftrag versendet"></transition>
  </task-node>
  <end-state name="Auftrag versendet" />
</process-definition>
```

## Collaboration Common Language

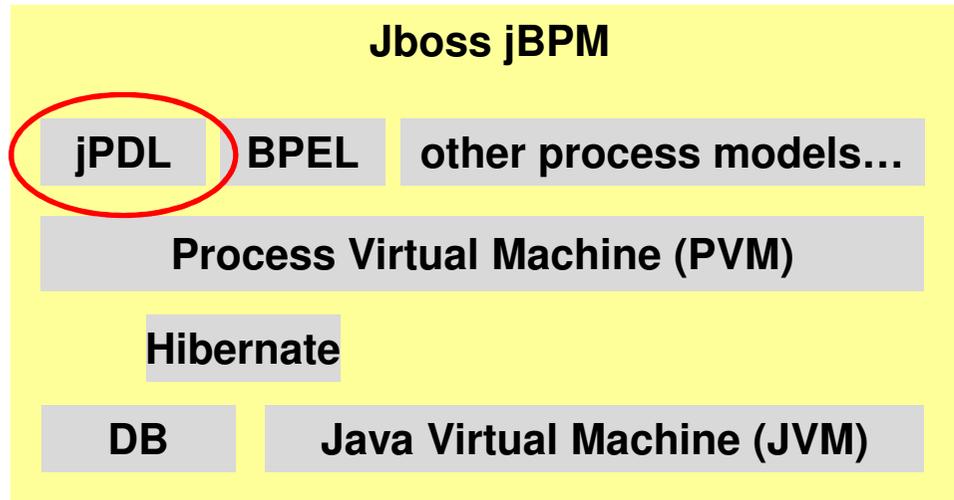


# jPDL vs. BPEL



- Anerkannter Standard
- Transaktionssteuerung nur auf Ebene der Services
- Alles asynchron
- WSDL

→ Ist BPEL vielleicht Service-Orchestrierung, nicht Business Process Management?



JBoss jBPM ist

- Business Process Engine
- „platform for process languages“

- Engine, die lang laufende Geschäftsprozesse ausführen kann
- Prozessinstanzen werden in DB gespeichert um langlebige Prozesse zu ermöglichen
- Wait-States
- Grafische Prozessdarstellung

# jPDL: Java Process Definition Language **camunda**

- Prozessbeschreibungen als XML-File
- Eigener grafischer Editor als Eclipse-Plugin
- Diagramme sind UML-Activity Diagrammen ähnlich
- Prozesse können externe Aktivitäten über Java-Actions ausführen
- Java-Objekte können als Prozessvariablen gespeichert werden

# Grafischer Prozessdesigner

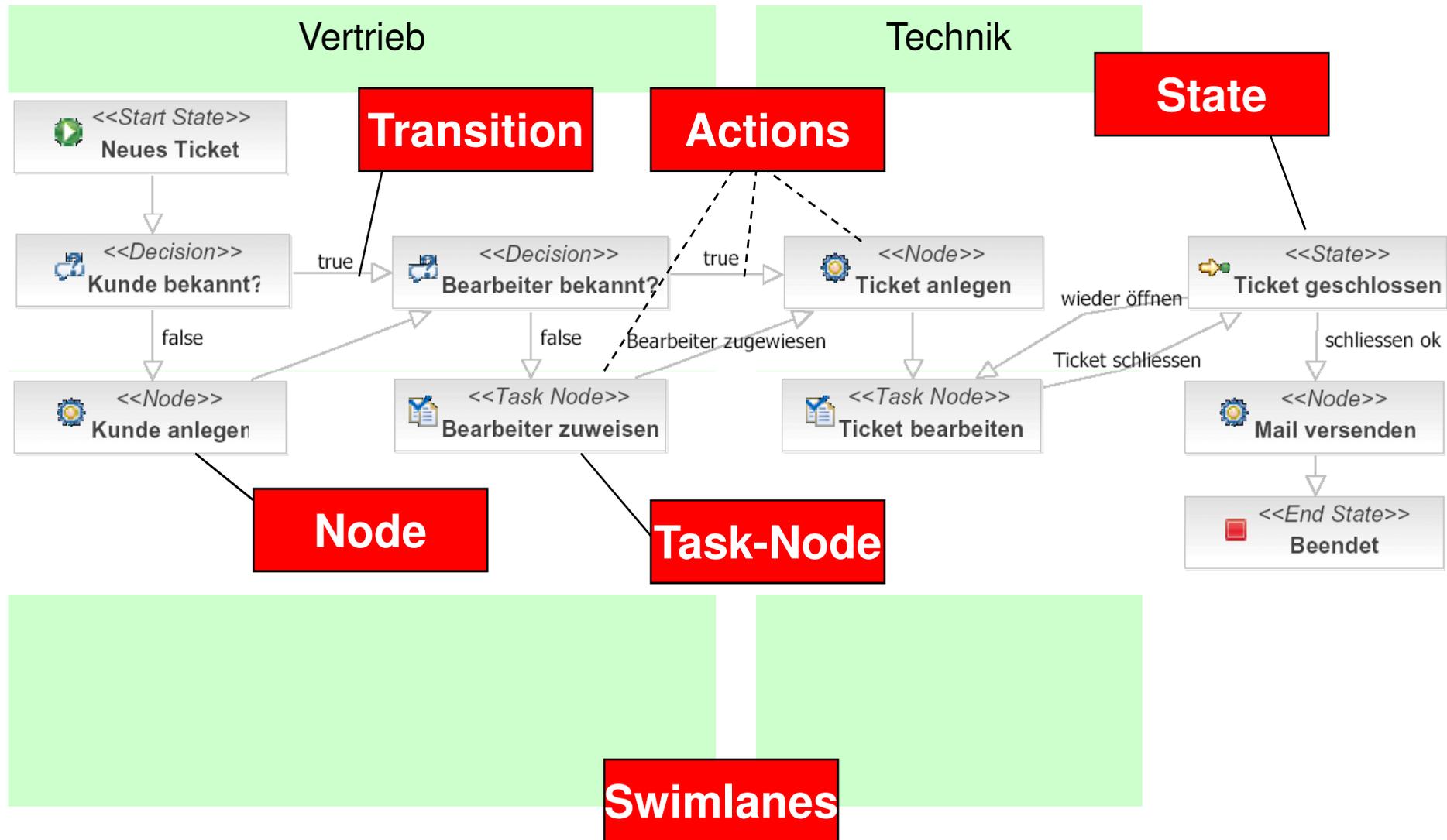
The screenshot shows the Camunda BPMN Designer interface within JBossIDE for Eclipse. The main workspace displays a state machine diagram for a process named 'SimpleOrder'. The diagram starts with a state 'Auftragseingang' (Start State), followed by a transition 'Auftrag anlegen' to a state 'Warten auf Zahlung' (State). From 'Warten auf Zahlung', there are two outgoing transitions: 'Zahlung OK' leading to a task node 'Auftrag versenden' (Task Node), and 'Zahlung nicht korrekt' leading to a self-loop on the 'Warten auf Zahlung' state. The 'Auftrag versenden' task node has an outgoing transition 'erledigt' leading to an end state 'Auftrag versendet' (End State).

The left sidebar shows the project hierarchy for 'JavaMagazinJbpmGuiSample', with the 'SimpleOrder' process definition file selected. The right sidebar shows the Outline view, displaying the structure of the 'SimpleOrder' process, including states and transitions.

The Properties view at the bottom shows the properties of the selected task node:

Property	Value
Class	com.camunda.jmgui.actions.CreateOrderAction
Name	create order in business logic

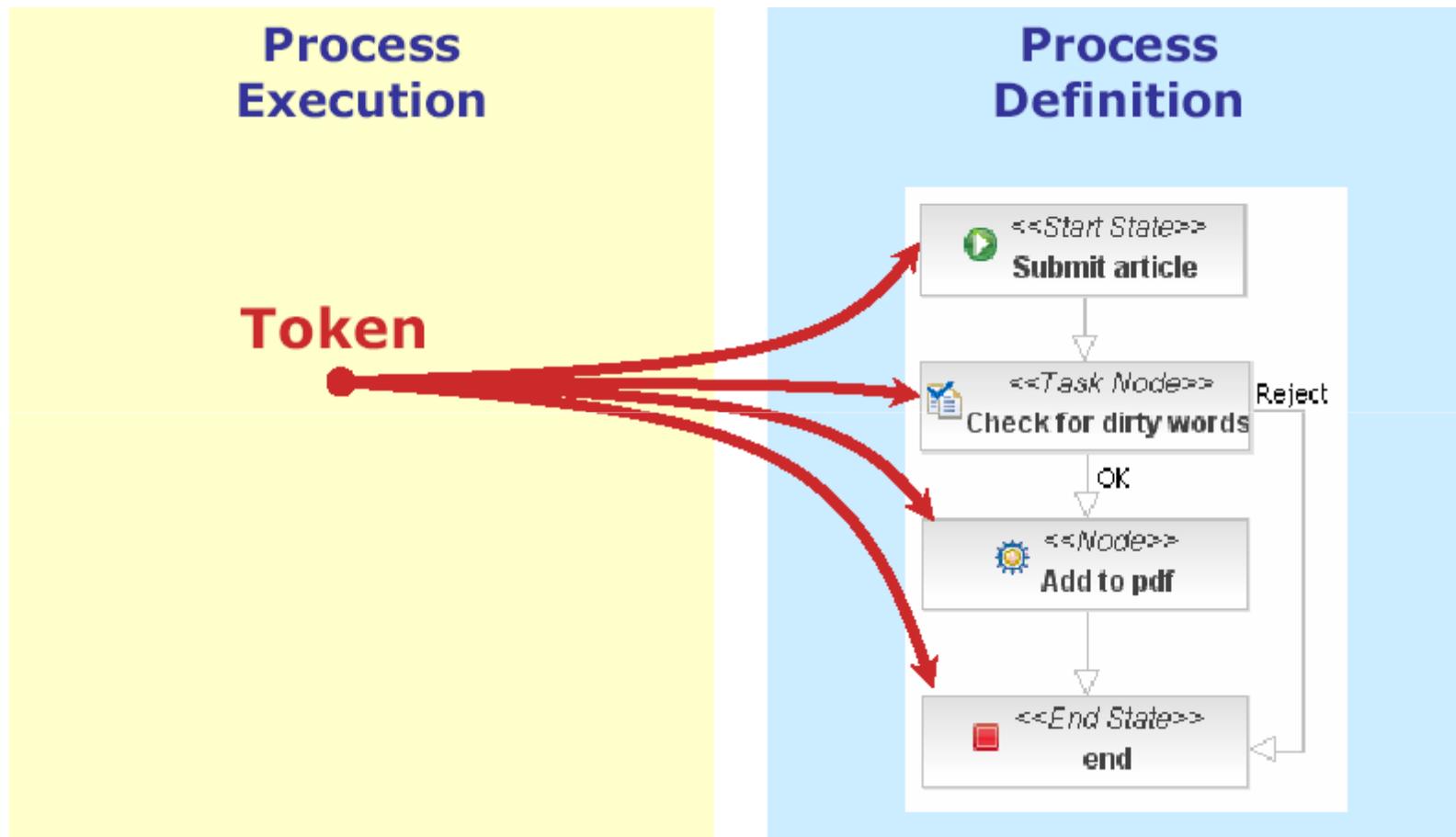
# Begriffe



- Task-Node
  - Warten auf menschliche Interaktion
  - Aufgabenliste / Todo-Liste / Worklist
- State
  - Warten auf externes Ereignis
  - Meist bei asynchroner Ansteuerung von Fremdsystemen
  - Keine Aufgabeliste!
- Node
  - Pseudo-Zustand
  - Visualisierung im Prozess
  - Ansatz für eigene Implementierungen

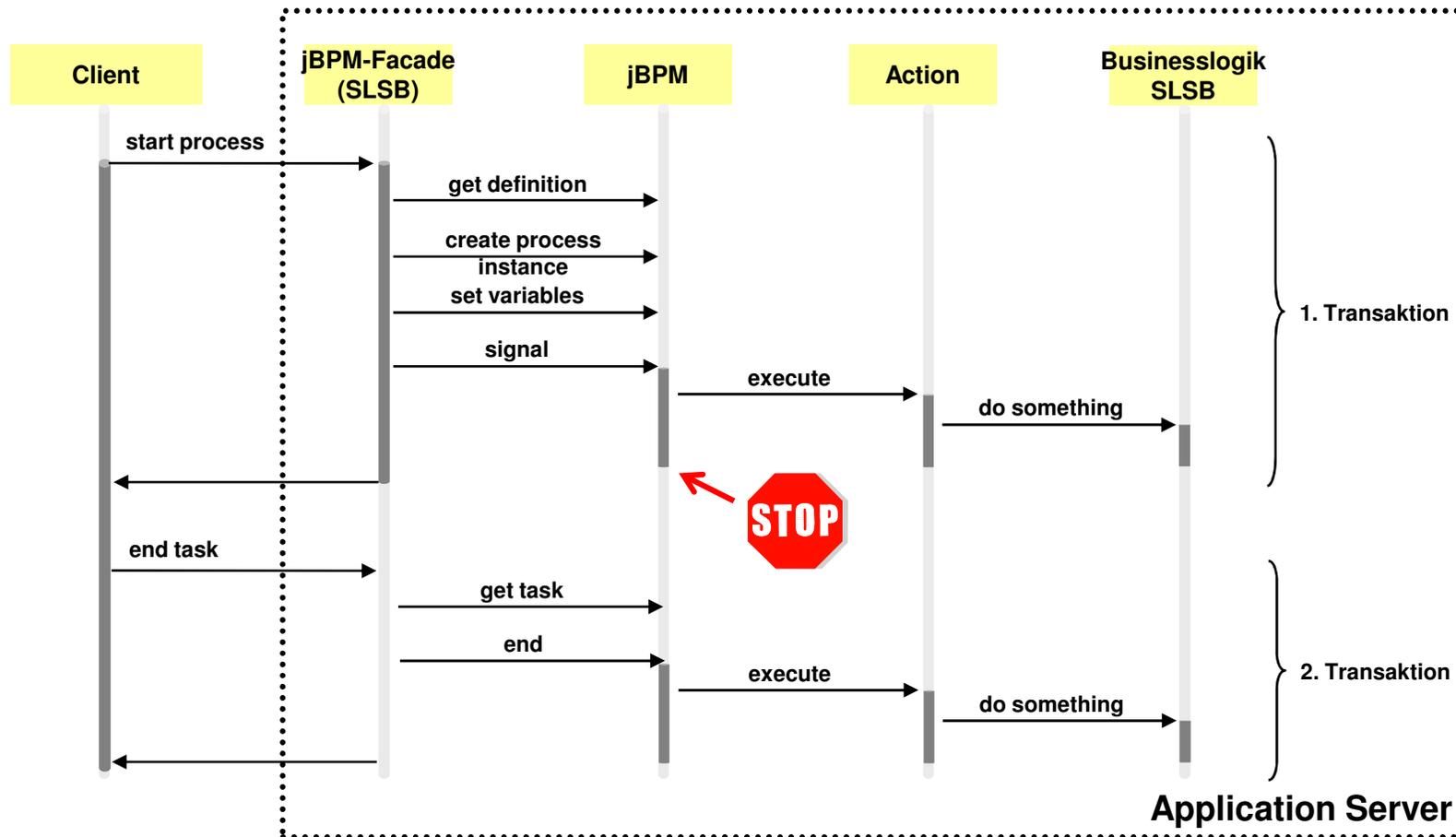
# Wartezustände





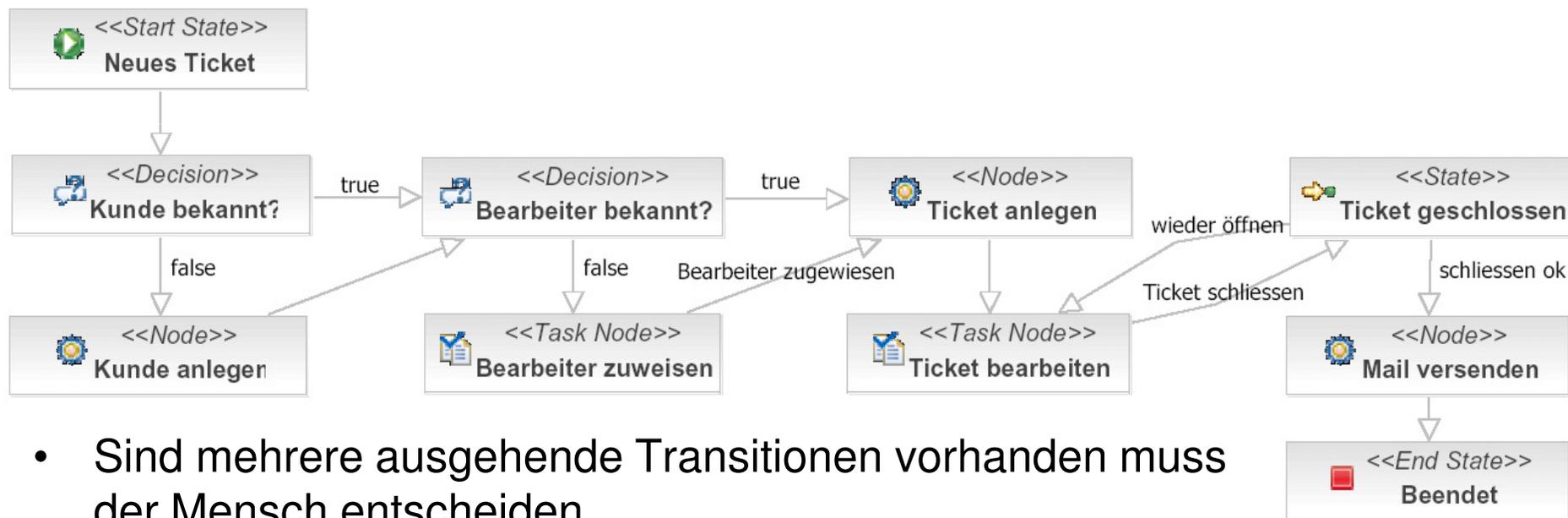
Quelle: Tom Baeyens

# Sequenzdiagramm



# jPDL 2 – Task-Nodes

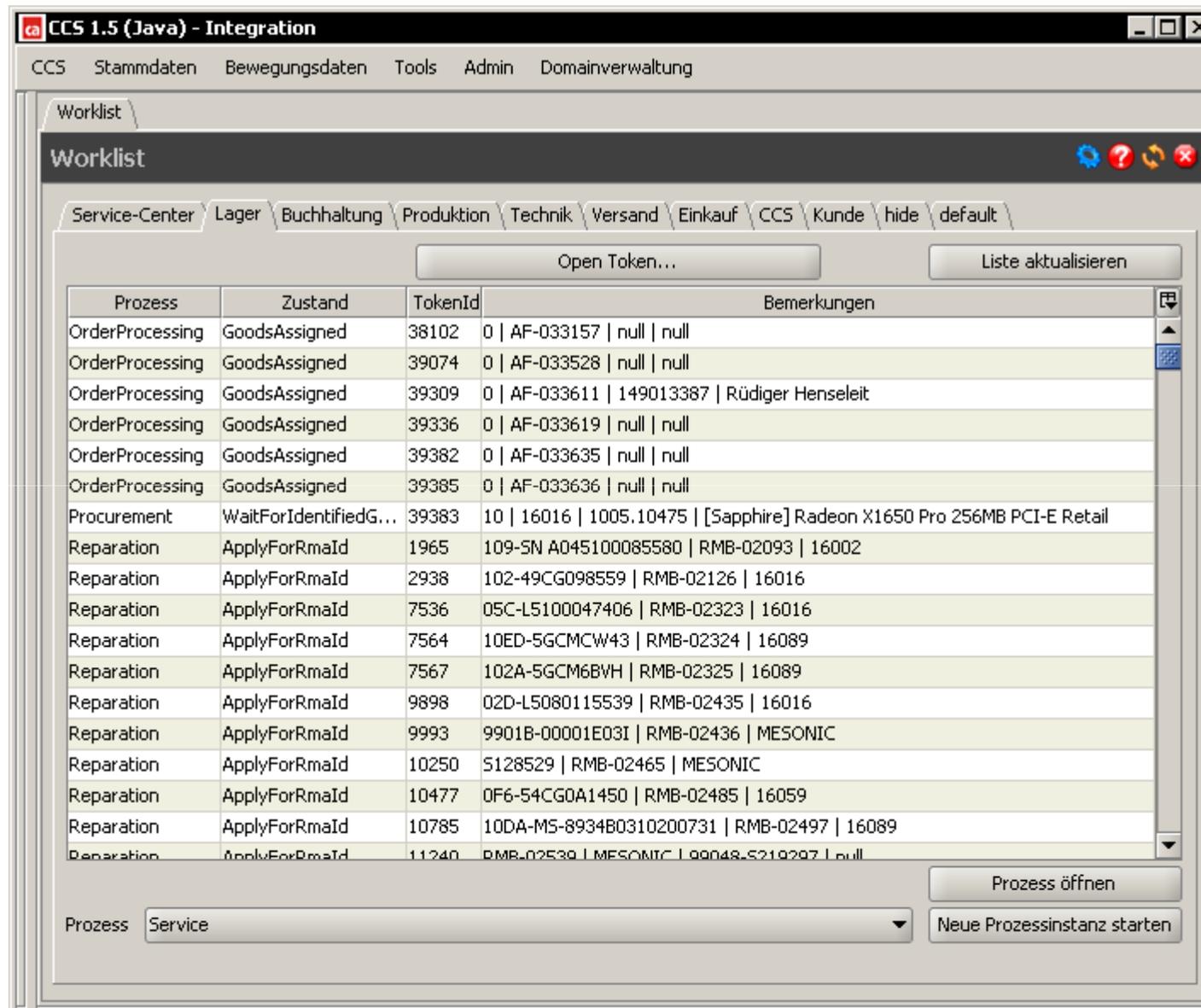
```
<task-node name="Bearbeiter zuweisen">
  <task swimlane="Vertrieb"/>
  <transition name="Bearbeiter zugewiesen" to="Ticket anlegen" />
</task-node>
<task-node name="Bearbeiter zuweisen">
  <task>
    <assignment expression="#{clerk}" />
  </task>
  <transition name="Bearbeiter zugewiesen" to="Ticket anlegen" />
</task-node>
```



- Sind mehrere ausgehende Transitionen vorhanden muss der Mensch entscheiden
- Ist kein Task vorhanden verhält sich die Task-Node wie eine Node

- In Task-Nodes werden „Tasks“ angelegt und einer Swimlane zugewiesen
- Abfragen der offenen Tasks durch eigene API möglich
- Bei Beendigung eines Tasks wird auch der State durch die ausgehende Transition verlassen

# Java Swing GUI: Generische Worklist



# Java Swing GUI

The diagram illustrates the mapping between a BPMN process and its Java Swing GUI implementation. The BPMN process on the left shows a task node 'visual inspection of customer data' with outgoing transitions 'cancel order' and 'data OK', and an incoming transition 'email changed'. This task is connected to a decision node 'method of payment?' which has outgoing transitions 'cod' and 'payment i'. The 'cod' transition leads to another task node 'wait for'. The Java Swing GUI on the right is a window titled 'CCS 1.5 - Produkt...' with a menu bar and a toolbar. The main content area displays a form for 'Order Processing / Visual Inspection Of Customer Data 34460'. The form contains fields for 'Anrede' (Herr), 'Vorname' (Bernd), 'Nachname' (Rücker), 'Firma' (camunda GmbH), 'Adresse 1' (Beim Oelsteg 10), 'Adresse 2', 'PLZ' (97980), 'Ort' (bad Mergentheim), 'Land' (DE), 'Provinz/Staat', 'Telefon', 'Mobil', and 'Telefax'. A 'Kopiere an...' button is located below the form. At the bottom right, there is a summary table with columns for labels and values, and a 'dataOk' button.

items	23,19	EUR
delivery costs	8,62	EUR
vat	5,09	EUR
sum_gross	36,90	EUR

Annotations in red text and arrows:

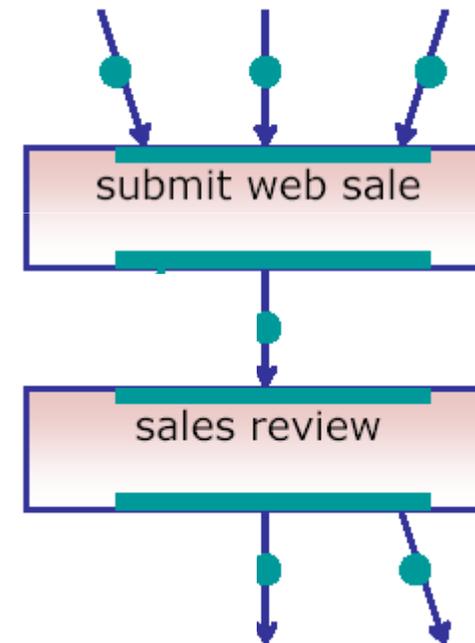
- node-name** and **token-id**: Point to the BPMN task node.
- documentation website**: Points to the browser address bar in the GUI screenshot.
- process data / variables**: Points to the form fields in the GUI.
- possible transitions**: Points to the 'dataOk' button in the GUI.

```
public class CreateCustomer implements ActionHandler {  
  
    public void execute(ExecutionContext ctx) throws Exception {  
        // get reference to CustomerService from JNDI  
        CustomerService services = (CustomerService)  
            new InitialContext().lookup("customerService");  
  
        // get variable "customer" out of process context  
        Customer customer = (Customer)ctx.getVariable("customer");  
  
        // create customer in CRM system  
        customer = services.createCustomer(customer);  
  
        // overwrite customer data in process context, because we have a id now  
        ctx.setVariable("customer", customer);  
    }  
}
```

**Aufruf eines Service**

Werden im Prozess platziert:

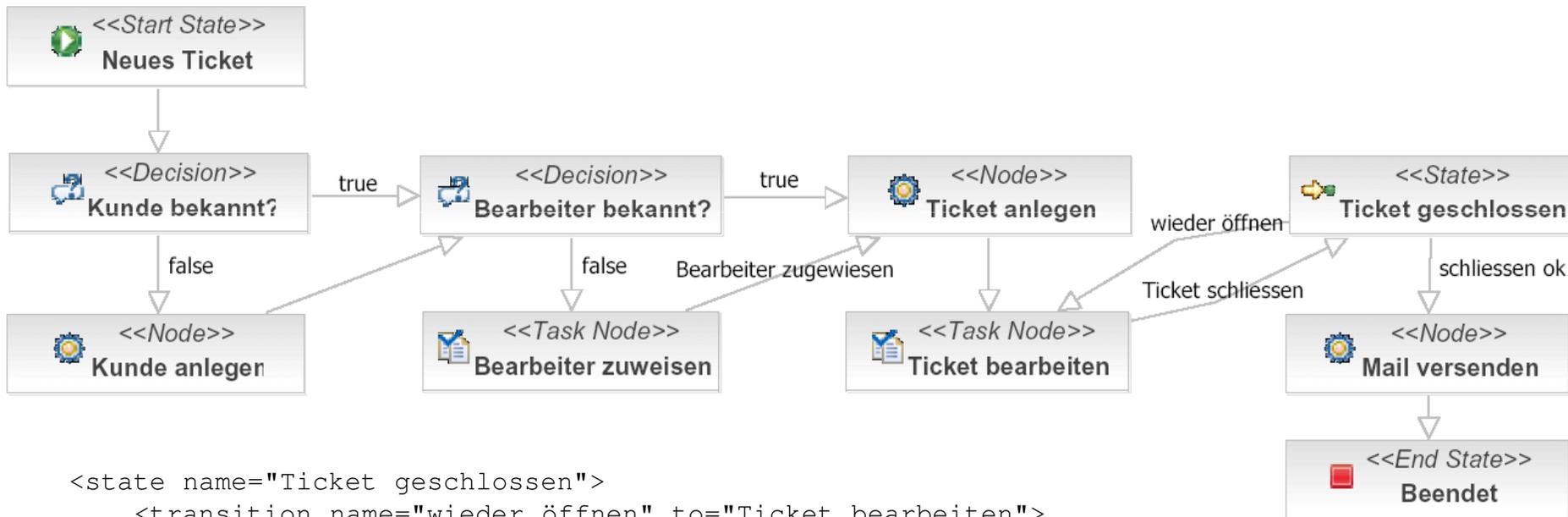
- Transitionen
- Node-Events
  - Node-Enter
  - Node-Leave



Quelle: Tom Baeyens

# Nodes & States

```
<node name="Kunde anlegen">
  <event type="node-enter">
    <action name="Kunde anlegen" class="de.ejbbuch.jbpm.actions.CreateCustomerAction"/>
  </event>
  <transition to="Bearbeiter bekannt?" />
</node>
```

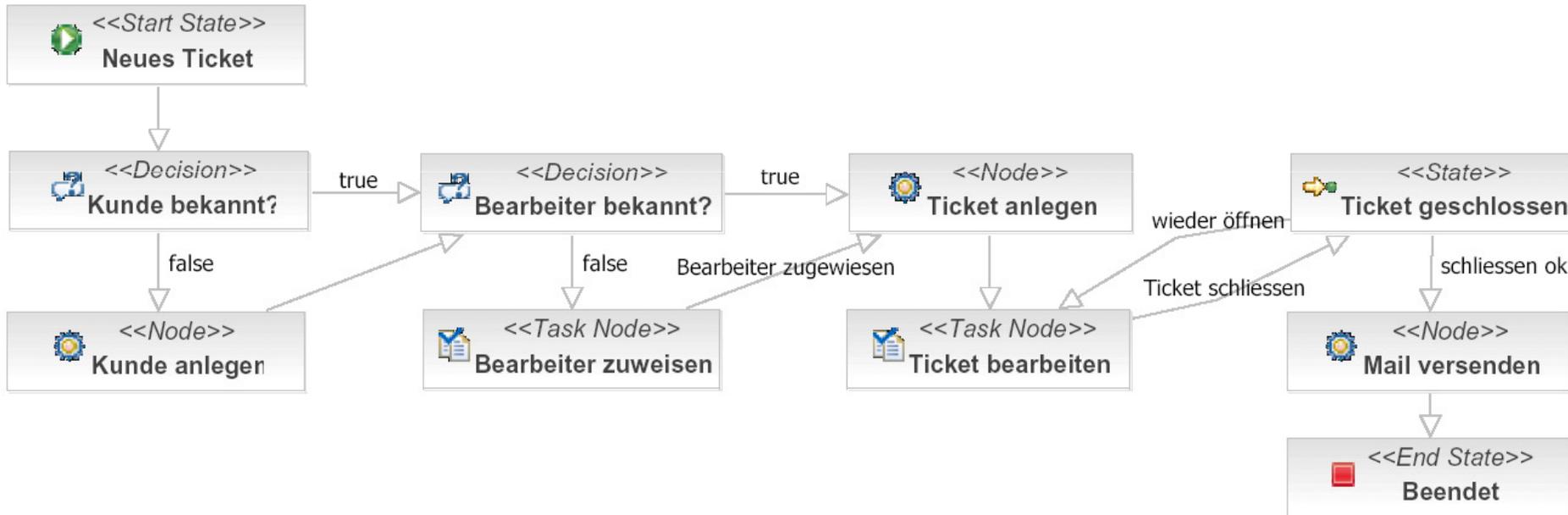


```
<state name="Ticket geschlossen">
  <transition name="wieder öffnen" to="Ticket bearbeiten">
    <action class="de.ejbbuch.jbpm.actions.ReopenTicketAction" />
  </transition>
  <transition name="schliessen ok" to="Mail versenden" />
</state>
```

# jPDL 2 – Decisions

```
<decision name="Kunde bekannt?">
  <handler class="de.ejbbuch.jbpm.decision.CustomerExistDecision"/>
  <transition name="false" to="Kunde anlegen" />
  <transition name="true" to="Bearbeiter bekannt?">
    <action class="de.ejbbuch.jbpm.actions.LoadCustomerAction" />
  </transition>
</decision>
```

Java



```
<decision name="Bearbeiter bekannt?" expression="#{clerk!=null}">
  <transition name="false" to="Bearbeiter zuweisen"></transition>
  <transition name="true" to="Ticket anlegen"></transition>
</decision>
```

Expression Language (EL)

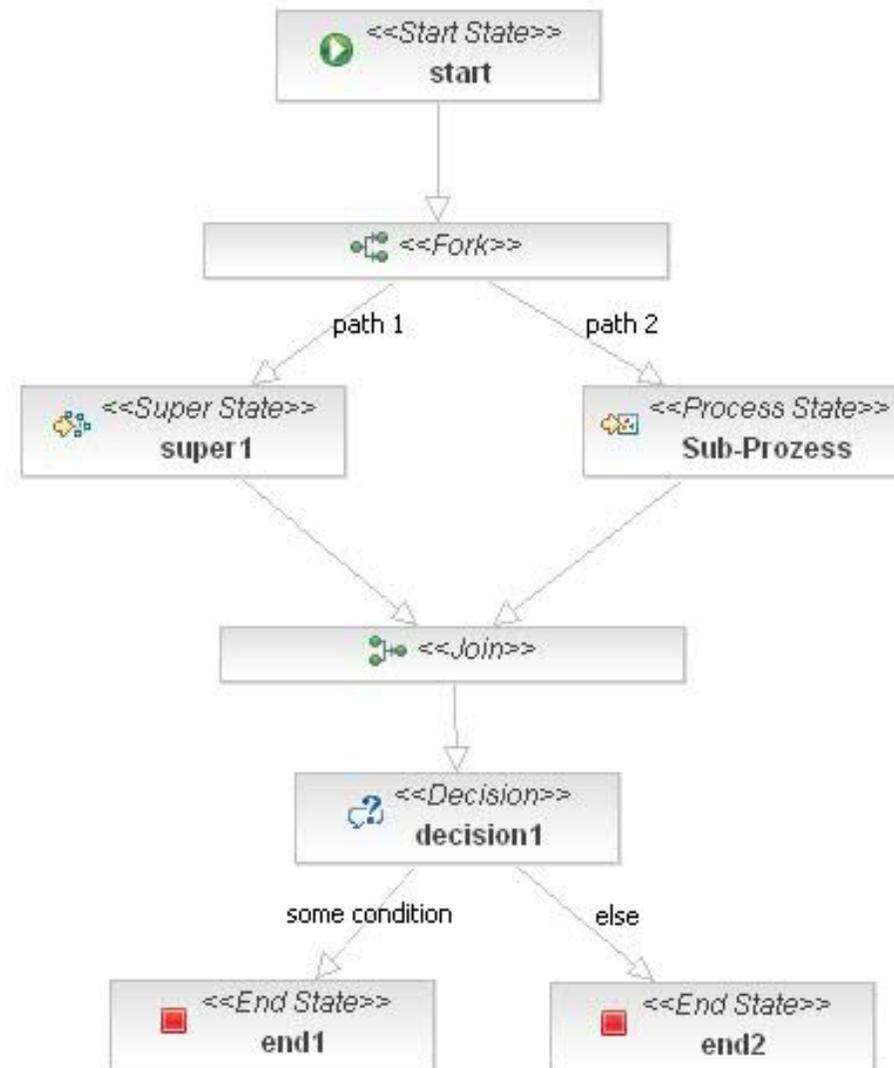
```
public class CustomerExistDecision implements DecisionHandler {

    public String decide(ExecutionContext ctx) throws Exception {
        // get variable "customer" out of process context
        Customer customer = (Customer)ctx.getVariable("customer");

        if (customer.getId() != 0)
            return "true";
        else {
            // get reference to CustomerServices
            CustomerService services =
                (CustomerService) new InitialContext().lookup("customerService");

            if (services.existCustomer(customer.getSurname(), customer.getGivenname()))
                return "true";
            else
                return "false";
        }
    }
}
```

- Timer
- Fork & join
- Subprozesse
- Super-States
- ...



- Sehr gut geeignet für „echtes“ BPM
  - Automatisierung
  - Task-Management
  - Integration
- Javaumgebung für Entwickler
- Gute Testbarkeit
  - Ohne Server / Container
  - Ohne Persistenz
  - ...

# Zugriff auf jBPM

```
JbpmConfiguration conf = JbpmConfiguration.getInstance();
JbpmContext context = conf.createJbpmContext();

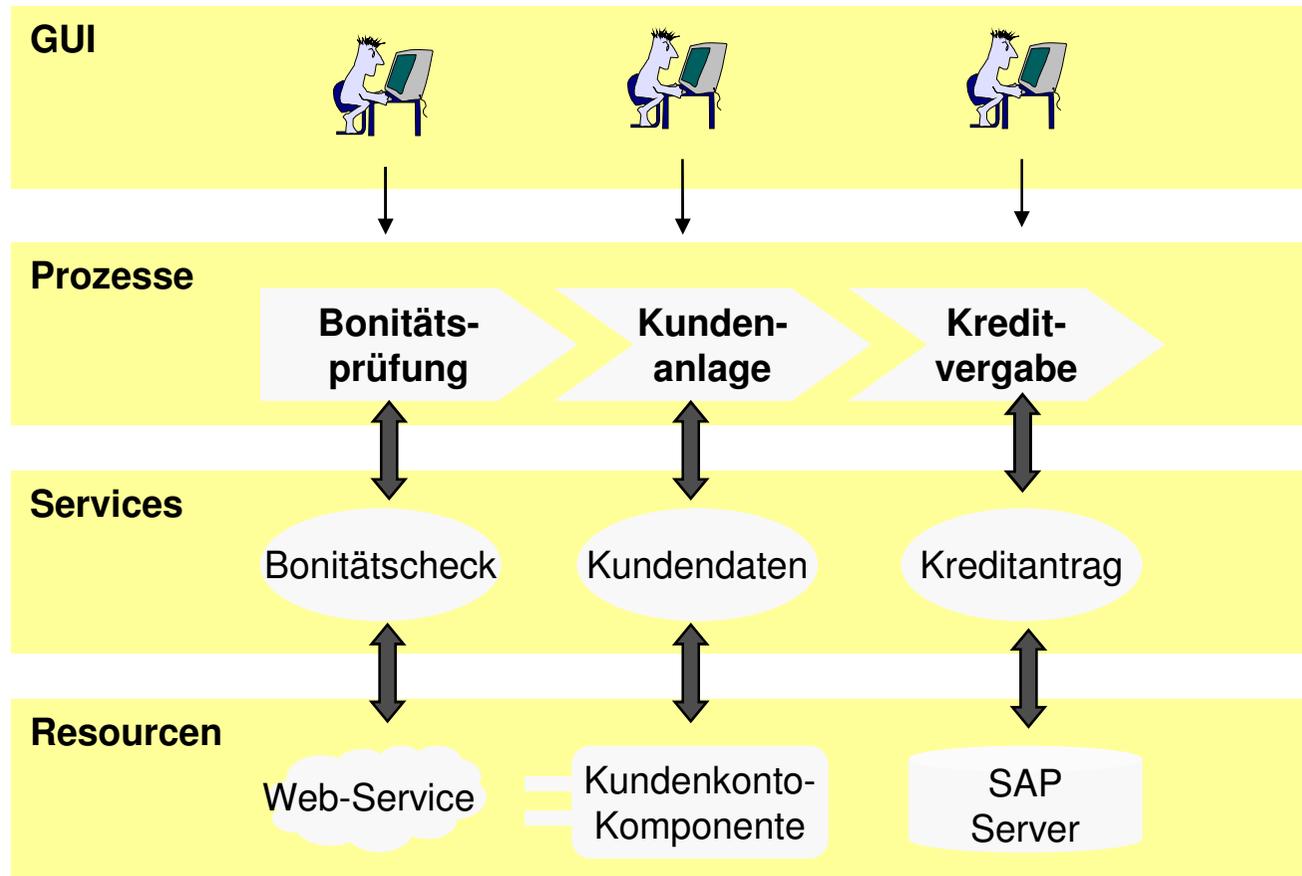
try {
    // Variablen für Prozess
    Map variables = new HashMap();

    // Prozess-Instanz "Ticket" starten
    ProcessInstance pi = context.getGraphSession()
        .findLatestProcessDefinition("Ticket")
        .createProcessInstance();
    pi.getContextInstance().addVariables(variables);

    pi.getRootToken().signal();

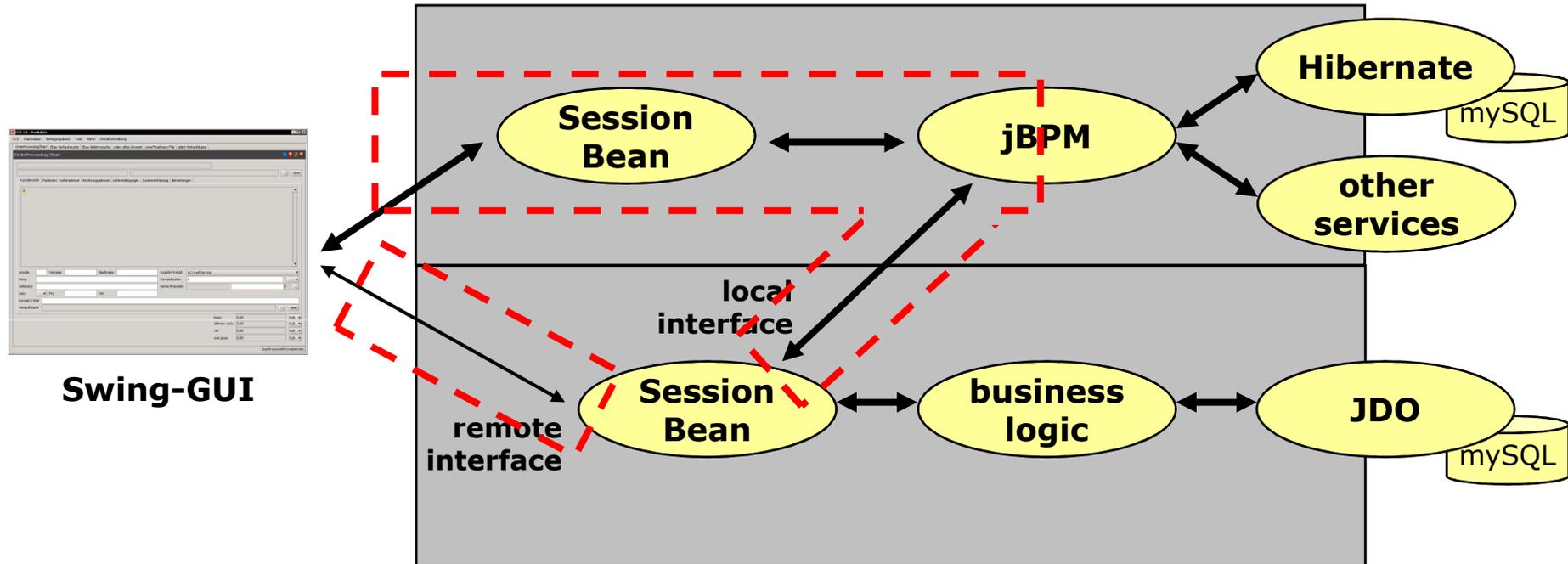
    // Alle Aufgaben für "Bernd Rücker"
    List<TaskInstance> tasks =
        context.getTaskMgmtSession().findTaskInstances("Vertrieb");

    // Transition "Ticket schliessen" ausführen
    tasks.get(0).end("Ticket schliessen");
}
finally {
    context.close();
}
```



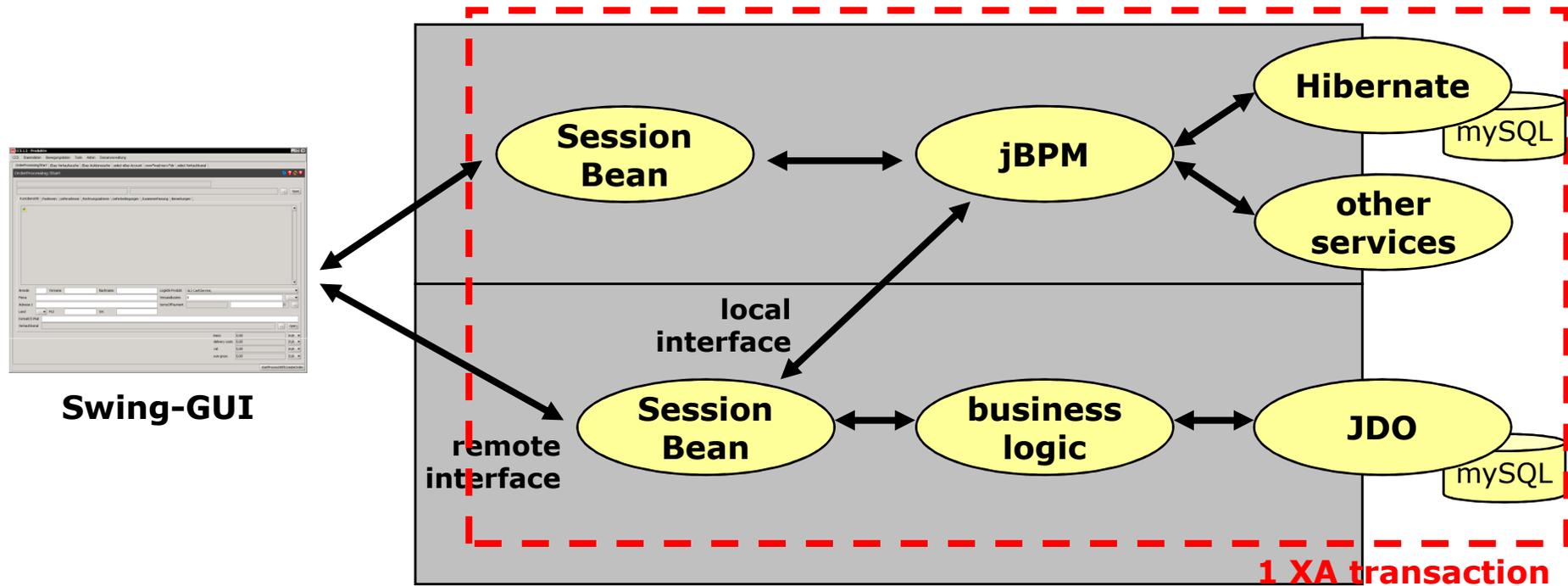
# Architecture - process is a facade!

- starting processes
- work on tasks



- Accessing / changing business objects directly
- Reports
- ...

# Architecture - transactions



- [www.camunda.com](http://www.camunda.com) → open source
- Eigenes Open Source Projekt zur Anbindung von Java Swing-Oberflächen
- Framework für User-Interaktion
- Admin-Client

The screenshot shows the Camunda Admin-Client interface. The title bar reads 'CCS 1.5 - Produktiv'. The main menu includes 'CCS', 'Stammdaten', 'Bewegungsdaten', 'Tools', 'Admin', and 'Domainverwaltung'. The 'jBPM Admin' window is active, displaying a sidebar with 'Aufgaben' (Tasks) and 'Administration' sections. The main area shows a 'ProcessInstance.list' view with filters for 'OrderProcessing' and a date range from 'Mo 11/13/2006'. A table lists process instances with columns for ID, name, version, token ID, node name, start date, and end date. An 'Aktualisieren' button is located above the table, and a 'Prozessinstanz öffnen' button is at the bottom right.

Prozess ID	Prozessname	Prozessversion	Wurzel-Token ID	Wurzel-Token Knotenname	Startdatum	Enddatum
34492	OrderProcessing	25	34497	VisualInspectionOfCustomerData	2006-11-13 19:33:57.0	
34491	OrderProcessing	25	34496	VisualInspectionOfCustomerData	2006-11-13 19:10:23.0	
34490	OrderProcessing	25	34495	VisualInspectionOfCustomerData	2006-11-13 19:01:33.0	
34489	OrderProcessing	25	34494	VisualInspectionOfCustomerData	2006-11-13 18:24:54.0	
34487	OrderProcessing	25	34492	VisualInspectionOfCustomerData	2006-11-13 17:55:27.0	
34484	OrderProcessing	25	34489	VisualInspectionOfCustomerData	2006-11-13 17:30:48.0	
34482	OrderProcessing	25	34487	WaitForPayment	2006-11-13 16:53:31.0	
34481	OrderProcessing	25	34486	WaitForPayment	2006-11-13 16:50:19.0	
34471	OrderProcessing	25	34476	VisualInspectionOfCustomerData	2006-11-13 16:47:57.0	
34460	OrderProcessing	25	34465	VisualInspectionOfCustomerData	2006-11-13 16:44:22.0	
34459	OrderProcessing	25	34464	WaitForPayment	2006-11-13 16:43:28.0	
34457	OrderProcessing	25	34462	VisualInspectionOfCustomerData	2006-11-13 16:21:50.0	
34456	OrderProcessing	25	34461	GoodsAssigned	2006-11-13 16:07:57.0	
34455	OrderProcessing	25	34460	VisualInspectionOfCustomerData	2006-11-13 16:06:07.0	
34450	OrderProcessing	25	34455	WaitForPayment	2006-11-13 15:58:43.0	

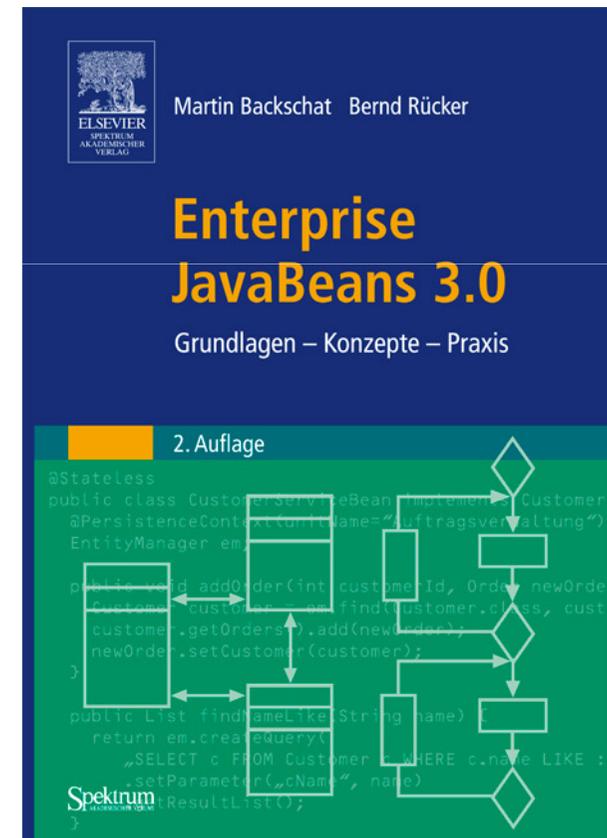
The screenshot shows the Camunda Admin-Client 2 interface. The main window displays the 'jBPM Admin' section for a process instance 'OrderProcessing 34484'. On the left, there are navigation menus for 'Aufgaben' (Tasks) and 'Administration'. The central area shows a tree view of the process instance, including a task 'VisualInspectionOfCustomerData' with a context menu open showing options like 'dataOk', 'emailChanged', and 'cancelOrder1'. Below this, a red box contains the text: 'tokens / tasks Commands: • Signal tokens • end tasks • cancel processes • ...'. On the right, the 'Detail-Informationen' section contains a 'jBpm-log' table and a 'Prozessvariablen' table. A red box labeled 'log' highlights the log table, and another red box labeled 'variables' highlights the process variables table. The log table has columns for Token ID, Datum, and Logeintrag. The process variables table has columns for Variablenname and Variablenwert.

Token ID	Datum	Logeintrag
34489	2006-11-13 17:30:48.0	node[decideKnownCustomer]
34489	2006-11-13 17:30:48.0	transition[decideKnownCustomer-->VisualIns...
34489	2006-11-13 17:30:48.0	action[MailServiceAction]
34489	2006-11-13 17:30:48.0	task-assign[null,TaskInstance[VisualInspectio...
34489	2006-11-13 17:30:48.0	task-create[TaskInstance[VisualInspectionOf...

Variablenname	Variablenwert
madmoxxNewsletter	false
orderValue	AF-031869;
driverNewsletter	false
orderKey	AF-031869;
__proc	-031869   null   null

- jbpm4jsf
- Neue Webconsole incl. Admin
- BPEL-Flavour wird stetig verbessert
- Eclipse-Plugin deutlich verbessert & erweiterungsfähig
- Simulationsumgebung auf Basis von jBPM, Master-Thesis ab 01.09.
  - Interesse?
- Offline-Fortführung von Geschäftsprozessen auf mobilen Geräten
  - Interesse?

- Betriebswirtschaftliche Literatur
- [www.jboss.com](http://www.jboss.com)
- [www.camunda.com](http://www.camunda.com)  
→ Publikationen
- EJB-Buch mit Integrationsteil:
  - Integration im Java Enterprise Umfeld
  - JBoss jBPM
  - BPEL mit Oracle



**camunda**  
The Business Process Company

**Vielen Dank für Ihre  
Aufmerksamkeit**



**[bernd.ruecker@camunda.com](mailto:bernd.ruecker@camunda.com)**