15.–18.09.2008
in Nürnberg

Herbstcampus

Wissenstransfer
par excellence

# Die Process Virtual Machine Hibernate des BPM?

Bernd Rücker

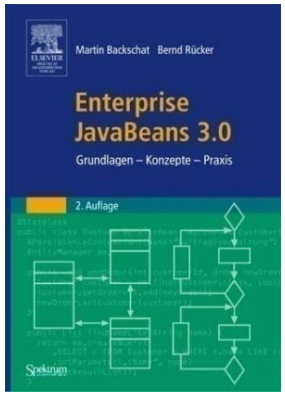
camunda services GmbH

# Bernd Rücker

camunda
The Business Process Company

Wer bin ich?

- Berater, Trainer, Coach
- Softwareentwickler
- Committer im JBoss jBPM-Projekt
- Themen: BPM, SOA, Process Execution (jBPM, BPEL, XPDL, …), Java EE
- Eigene Trainings zu Process Execution, BPMN, BPM-Software, …



# Agenda

camunda
The Business Process Company

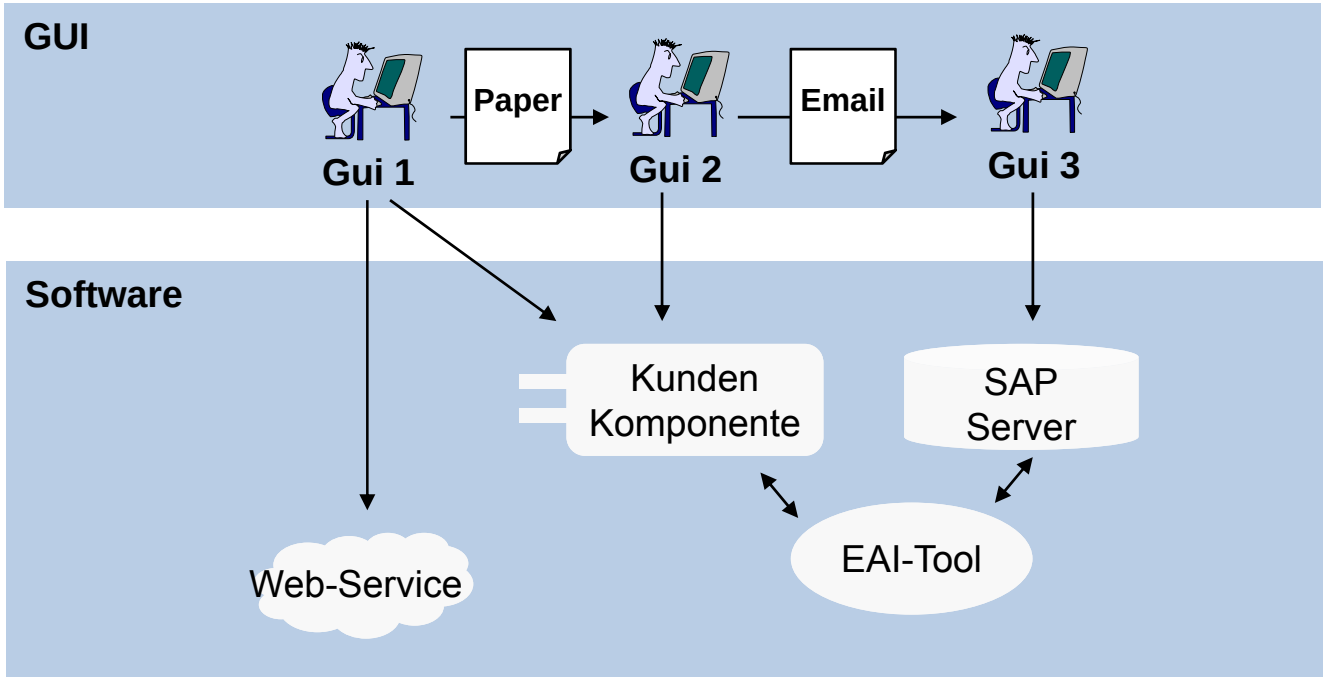
Geschäftsprozess und Regeln

- Einführung BPM (Business Process Management)
- Die Process Virtual Machine (PVM)
- jBPM jPDL 3
- Beispielprozess mit Demo
- Fazit

# Geschäftsprozesse & Software

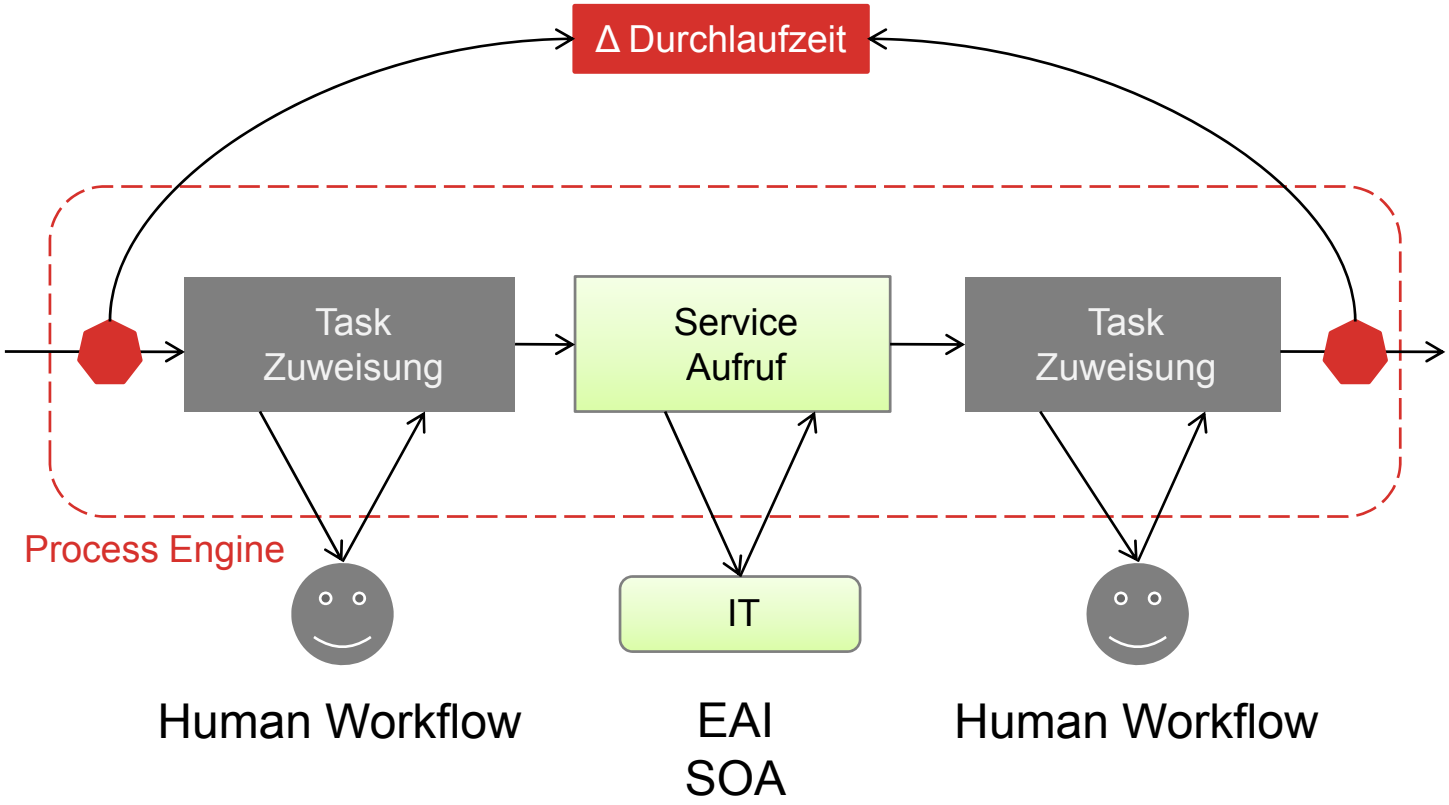
camunda
The Business Process Company

## Der Status Quo in manchen Unternehmen



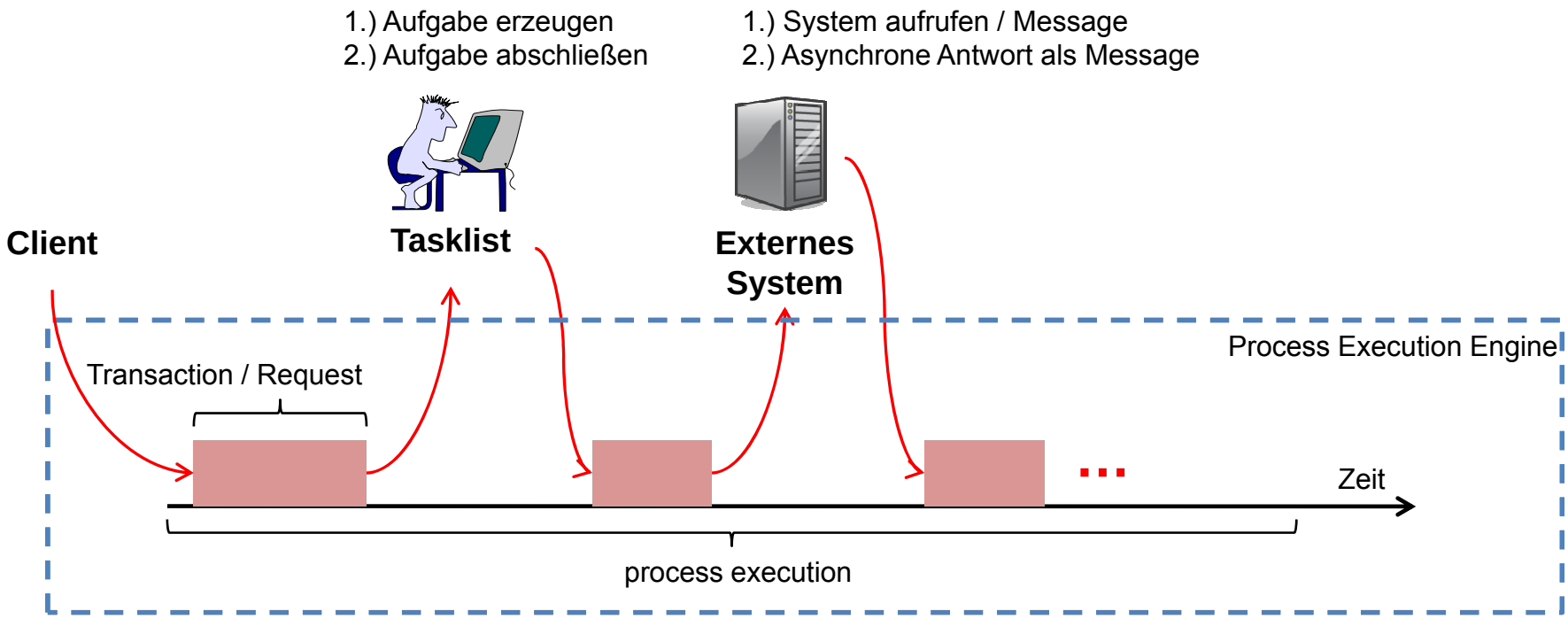
# Ein „digitaler“ Prozess

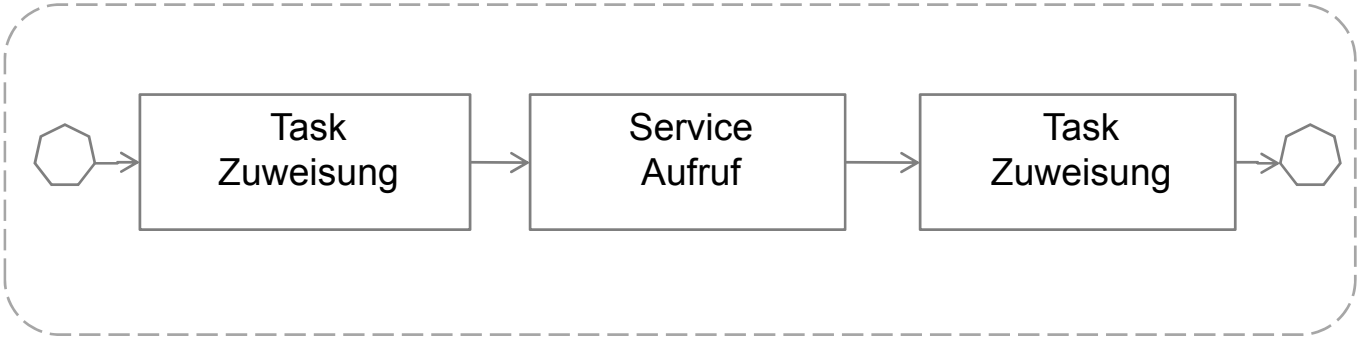
mit Business Process Engine



# Process Execution

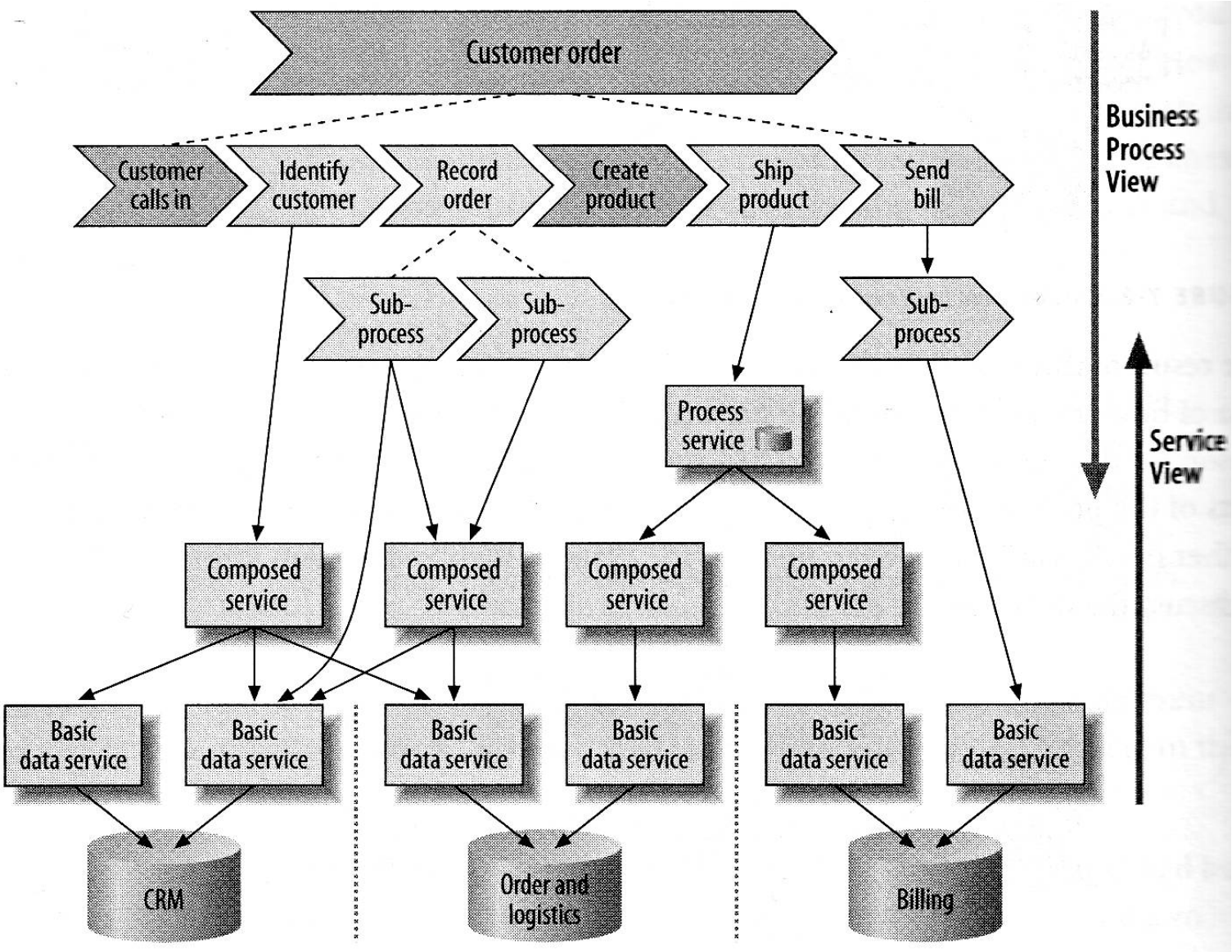
## Business Process Engine



# Integration: BPM & SOA

camunda
The Business Process Company

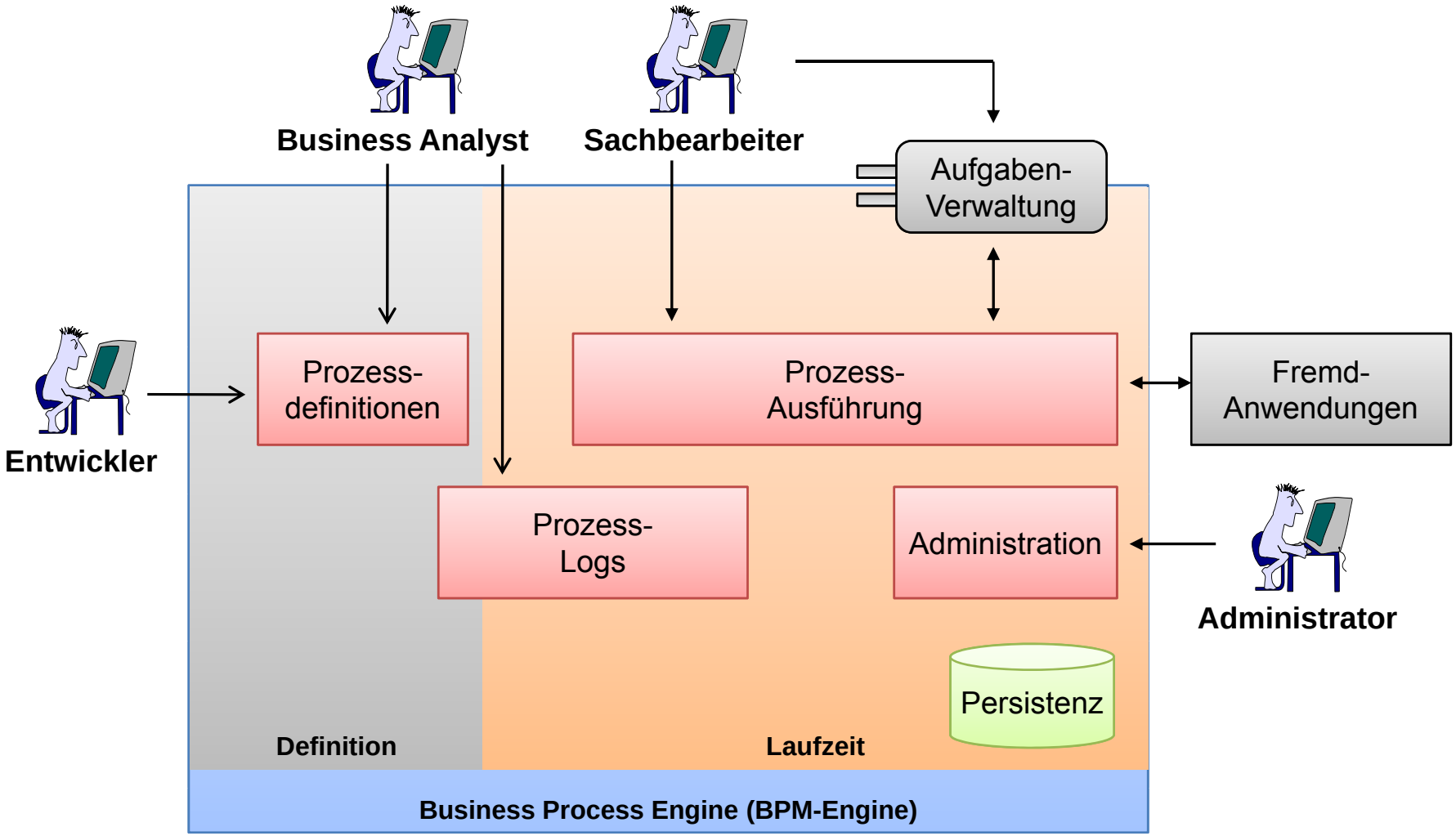



Quelle: Nicolai Josuttis

# Business Process Engine

camunda
The Business Process Company

## Middleware für Geschäftsprozesse



# Was leistet die Business Process Engine

camunda
The Business Process Company

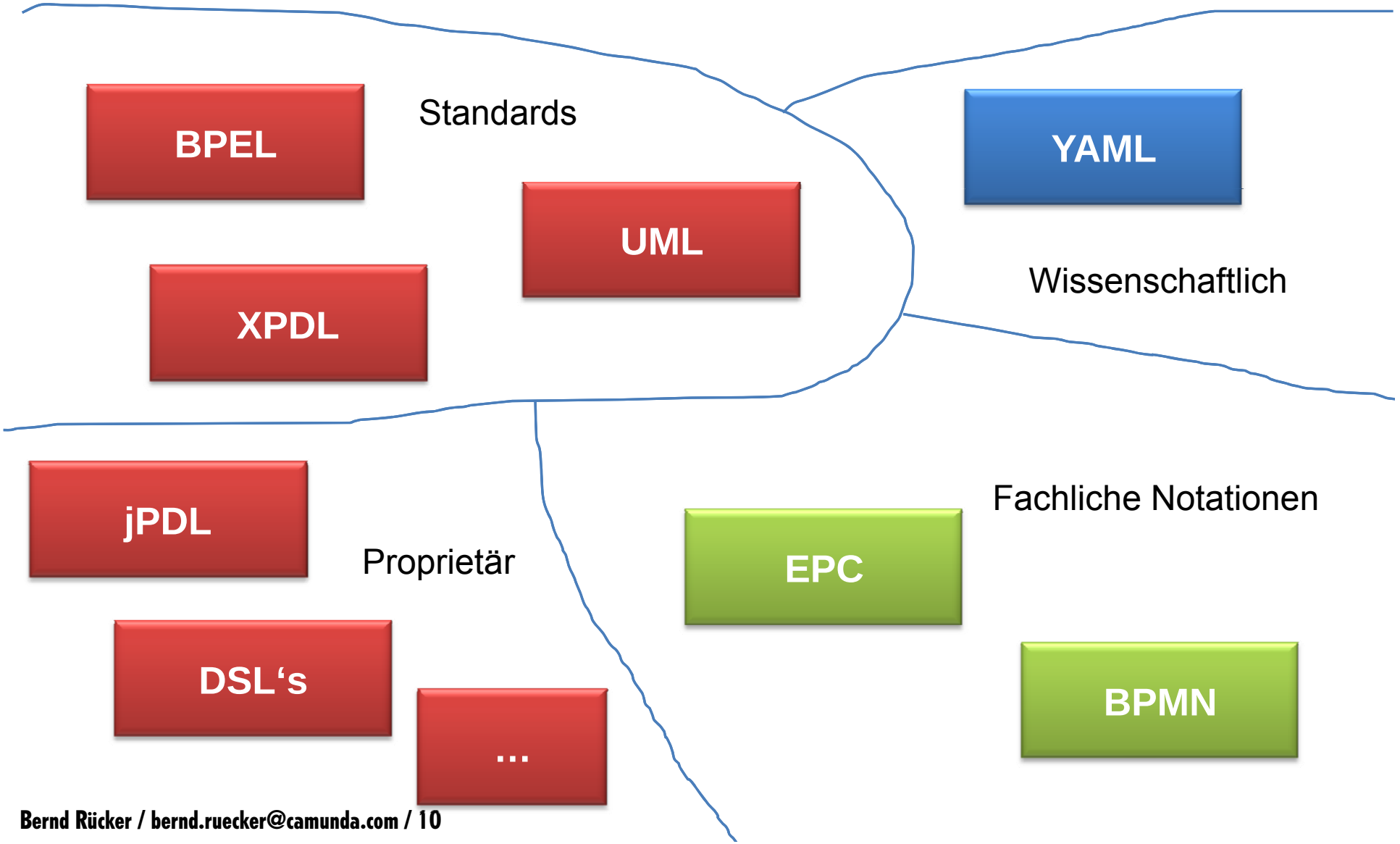
Features

- Versionierung, Persistenz & Interpretation von Prozessmodellen
- Steuerung & Persistenz von Prozessinstanzen
- Task-Management
- Wait-States
- Prozesskontext (Variablen zu Prozess speichern)
- Einbindung externer Services
- Verwalten von Ereignissen (z.B. Timeouts)
- …

# Process Execution Languages

camunda
The Business Process Company

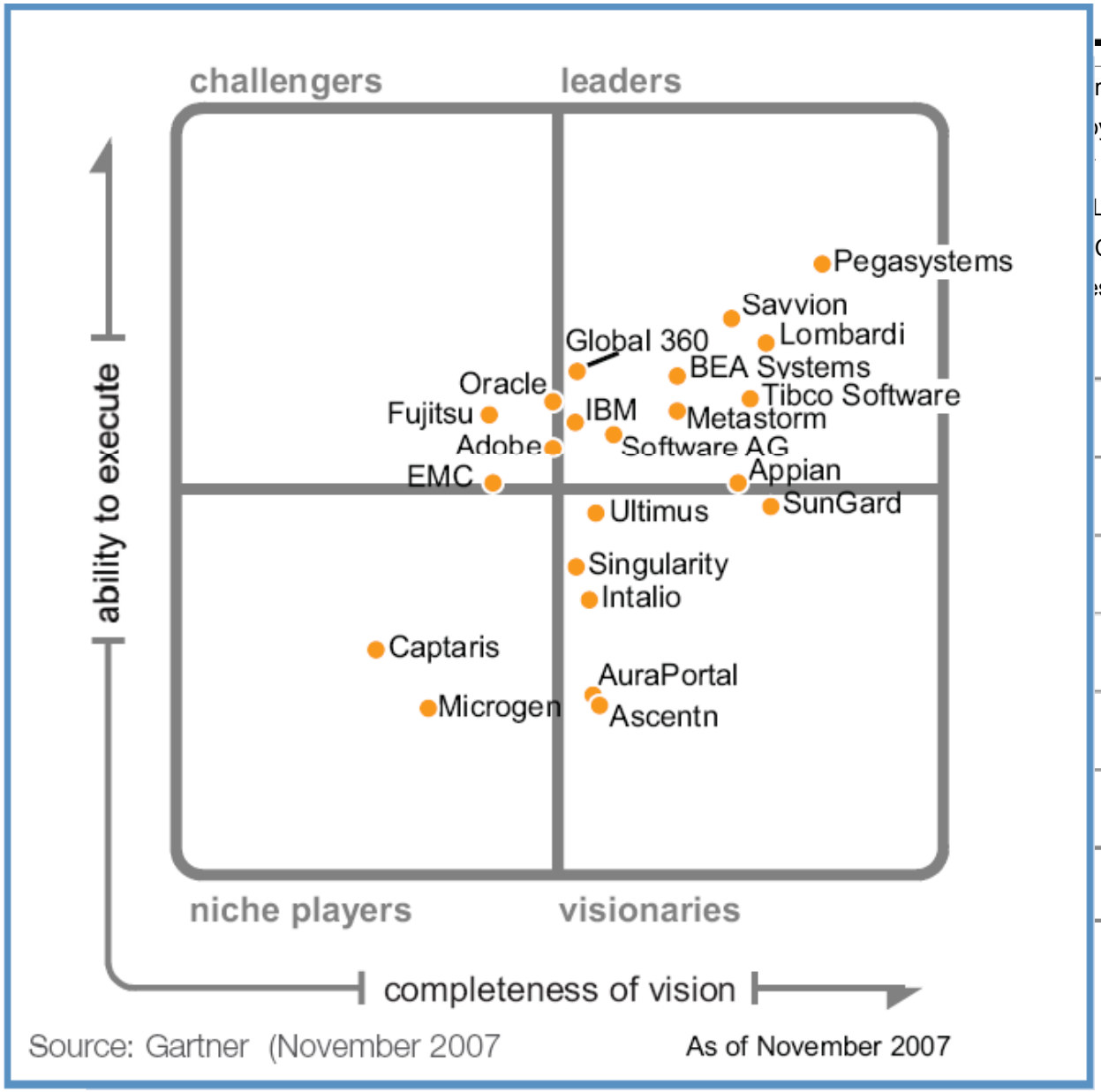
Welche Sprache spricht BPM?

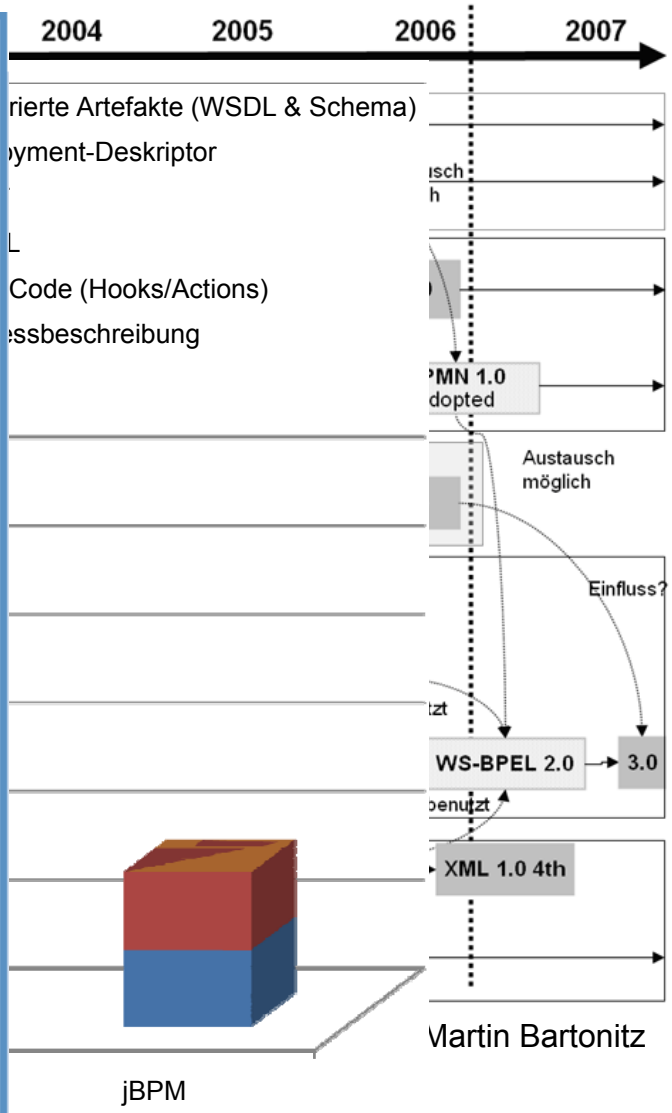


# Sprachen heute: Was nehmen?

camunda
The Business Process Company

Komplexität, LOC, Mächtigkeit, Standards?



# Motivation Process Virtual Machine

camunda
The Business Process Company

JBoss PVM

- Es existieren verschiedenste Prozessausführungs-sprachen (Process Execution Language)
  - BPEL, XPDL, jPDL, DSL‘s, …
- Es gibt nicht **die** perfekte Sprache
- Koexistenz von verschiedenen Sprachen erlauben
- Sprache nach Problem auswählen

→ Grundfunktionalität Prozessmaschine in PVM

# JBoss PVM

camunda
The Business Process Company

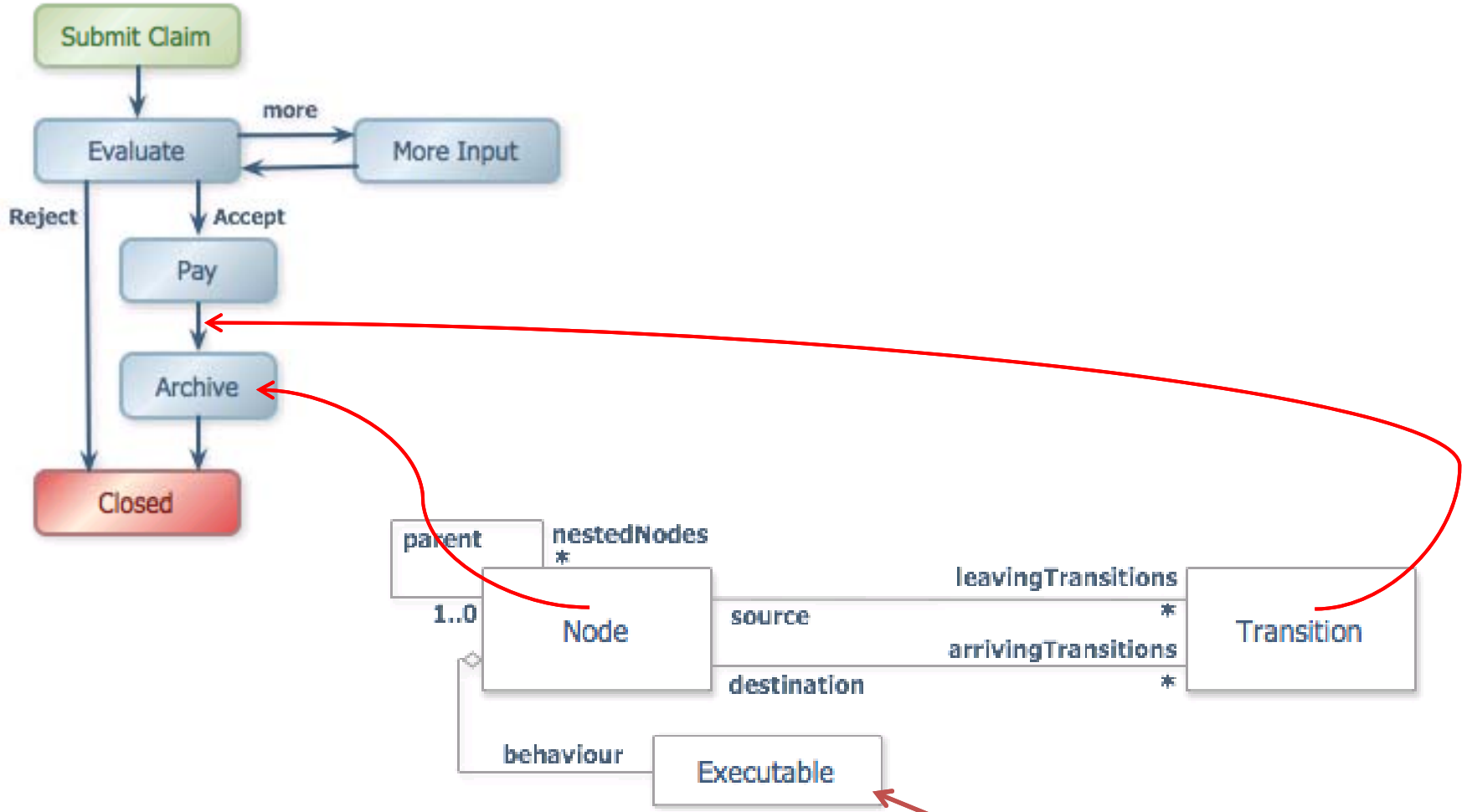
## Das Projekt

- Gestartet 2007 durch JBoss (jBPM) und Bull (Bonita & Orchestra)
- Gehostet bei JBoss, LGPL
- POJO-Kern: Interne Prozessrepräsentation durch Java-Modelle
- Persistenz austauschbar (Hibernate, EJB3)
- Lauffähig mit oder ohne Application-Server
- „Library“, kann eigebettet werden
- Aktuell Alpha Version

# Konzepte & Kernabstraktionen

camunda
The Business Process Company

## Prozesse als Zustandsautomat

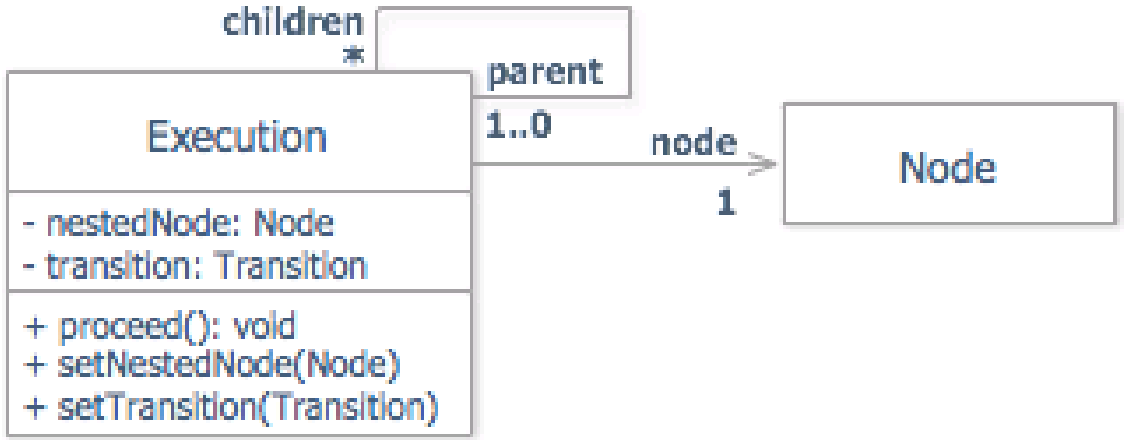


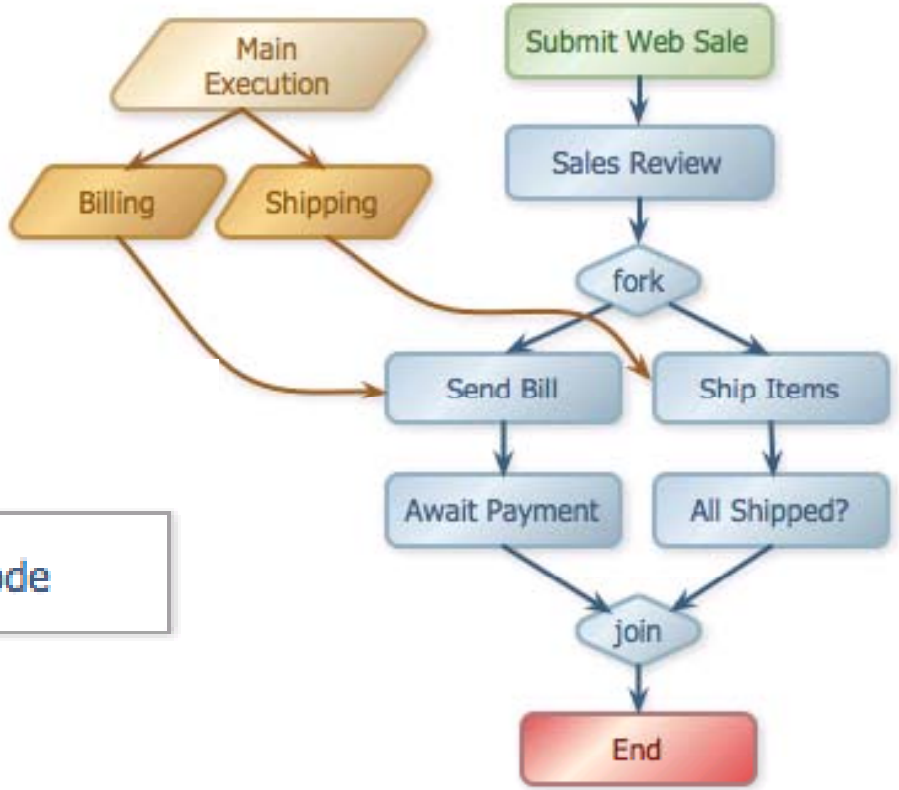

Verhalten über Sprache definiert

# Laufzeitverhalten

camunda
The Business Process Company

## Prozesse als Zustandsautomat

# Verhalten von Nodes

camunda
The Business Process Company

## Activity (entspricht Executable)

Einfaches Interface um Verhalten zu implementieren:

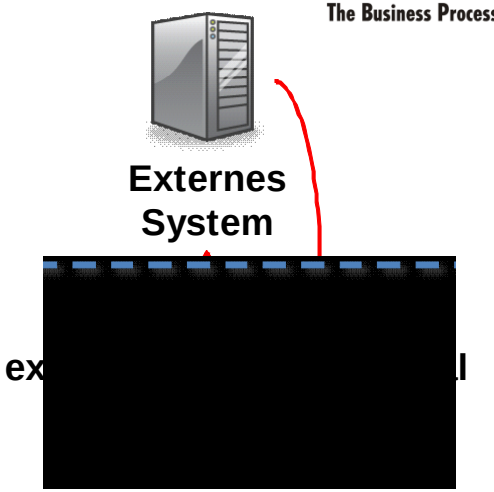
```
public interface Activity {
  void execute(Execution execution) throws Exception;
}
```

Beispiel:

```
public class Display implements Activity {

  String message;

  public Display(String message) {
    this.message = message;
  }

  public void execute(Execution execution) {
    System.out.println(message);
  }
}
```

# Verhalten von „externen Nodes“

## External Activity

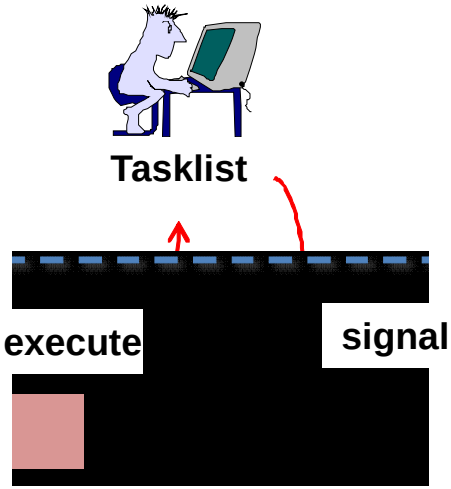



```
public class StateActivity implements ExternalActivity {

  public void execute(Execution execution) {
    execution.waitForSignal();
  }

  public void signal(Execution execution, String signalName,
                     Map<String, Object> parameters) {
    if (parameters!=null) {
      execution.setVariables(parameters);
    }
    execution.take(signalName);
  }
}
```

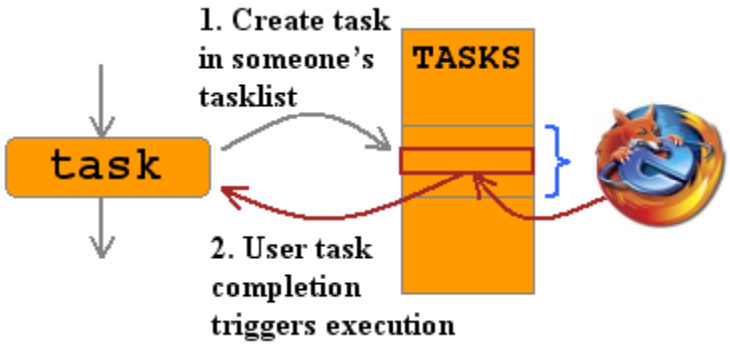
# Beispiel: TaskActivity

z.B. jPDL TaskNode oder BPEL PeopleActivity

camunda
The Business Process Company



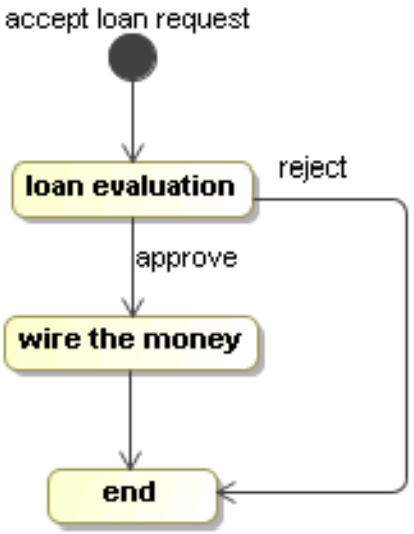

```
public class TaskActivity implements ExternalActivity {

  public void execute(Execution execution) {
    // let's use the node name as the task id
    String taskName = execution.getNode().getName();
    TaskComponent.createTask(taskName, execution);
  }

  public void signal(Execution execution, String signal,
                     Map<String, Object> parameters) {
    execution.takeDefaultTransition();
  }
}
```



# Just an API – PVM definiert keine Sprache

camunda
The Business Process Company

Ein erster kleiner Prozess

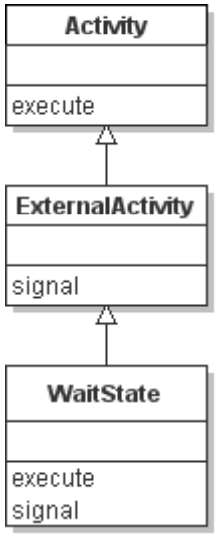



```
ProcessDefinition processDefinition = ProcessFactory.build()
    .node("accept loan request").initial().behaviour(new WaitState())
      .transition().to("loan evaluation")
    .node("loan evaluation").behaviour(new WaitState())
      .transition("approve").to("wire the money")
      .transition("reject").to("end")
    .node("wire the money").behaviour(new Display("automatic payment"))
      .transition().to("end")
    .node("end").behaviour(new WaitState())
.done();
```

# Implementierung Wartezustand

camunda
The Business Process Company

Beispielsweise der jPDL "State"




```
public class WaitState implements ExternalActivity {

  public void execute(ActivityExecution execution) {
    execution.waitForSignal();
  }

  public void signal(ActivityExecution execution,
                     String signal,
                     Map<String, Object> parameters) {
    execution.take(signal);
  }
}
```

# Events

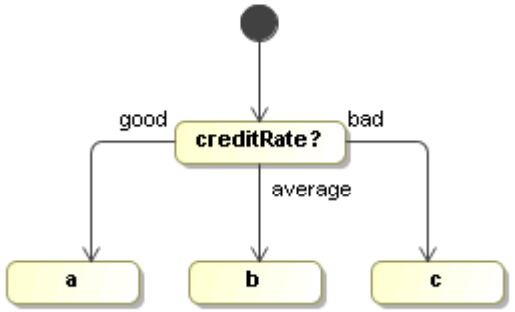
"Hooks" im Prozessablauf


```
public class Display implements Activity {
  String message;

  public Display(String message) {
    this.message = message;
  }
  public void execute(Execution execution) {
    System.out.println(message);
  }
}
```

- An Transitionen
- Node-Enter / Node-Leave
- Prozess-Start / Prozess-End

# Automatische Entscheidungen

"Nur" eine andere Art Activities



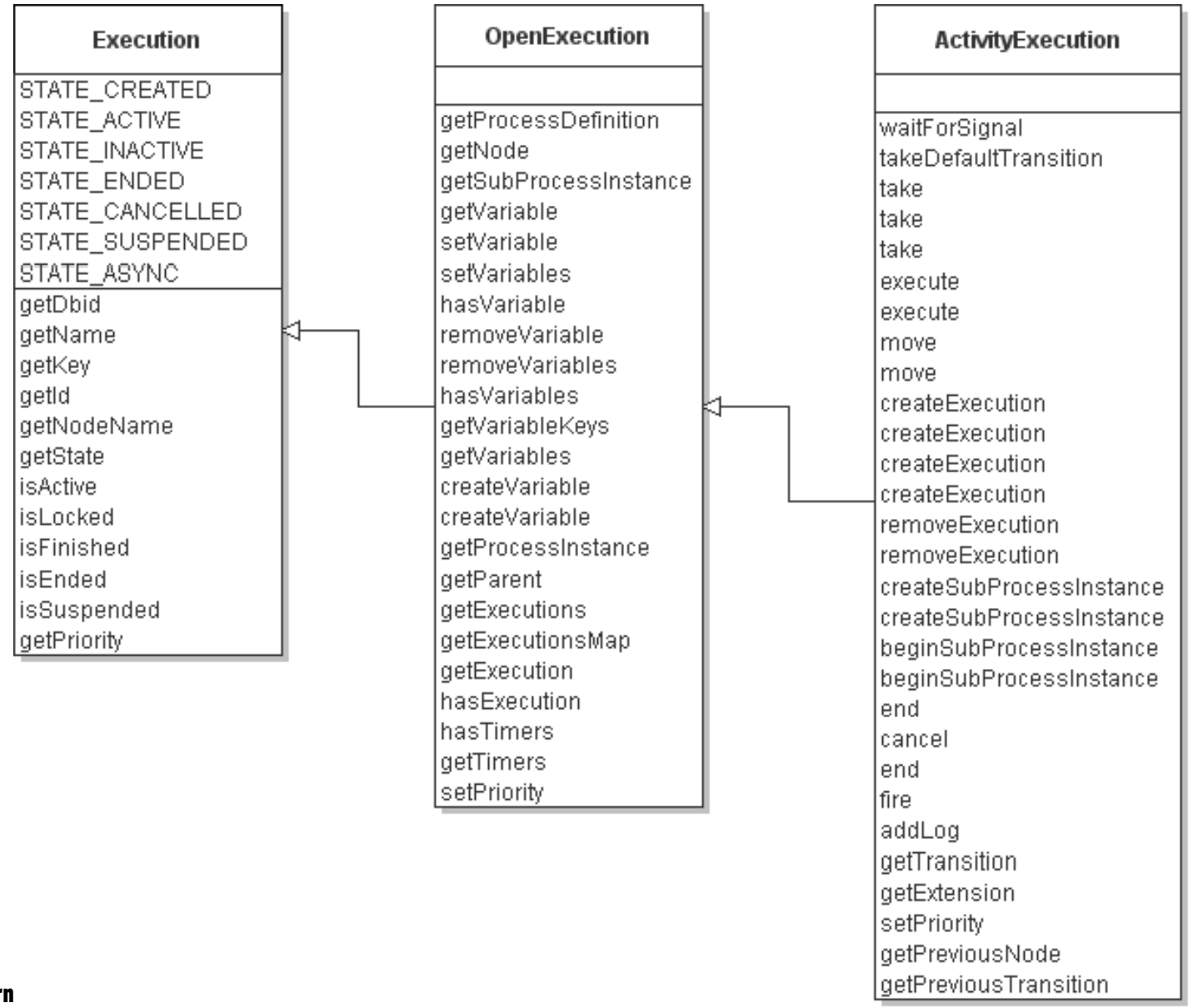

```
public class AutomaticCreditRating implements Activity {
  public void execute(Execution execution) {
    int creditRate = (Integer) execution.getVariable("creditRate");

    if (creditRate > 5)
      execution.take("good");
    else if (creditRate < -5)
      execution.take("bad");
    else
      execution.take("average");
  }
}
```

→ Durch Parametrisierung (gesteuert über die Sprache) kann jPDL- oder BPEL-Entscheidung gebaut werden

# Execution

camunda
The Business Process Company

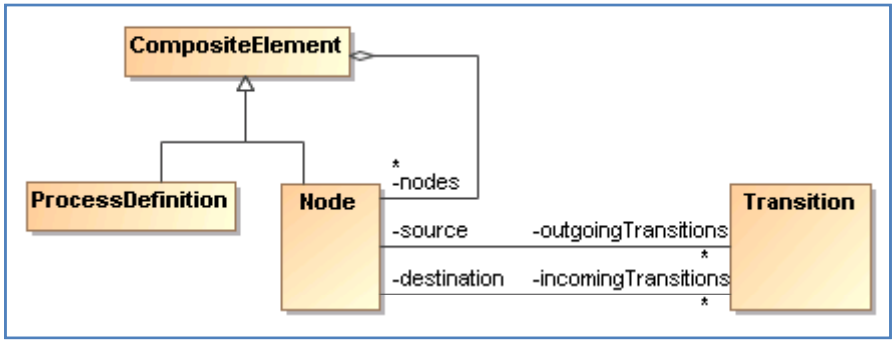
## Die Schnittstelle zur Prozessmaschine

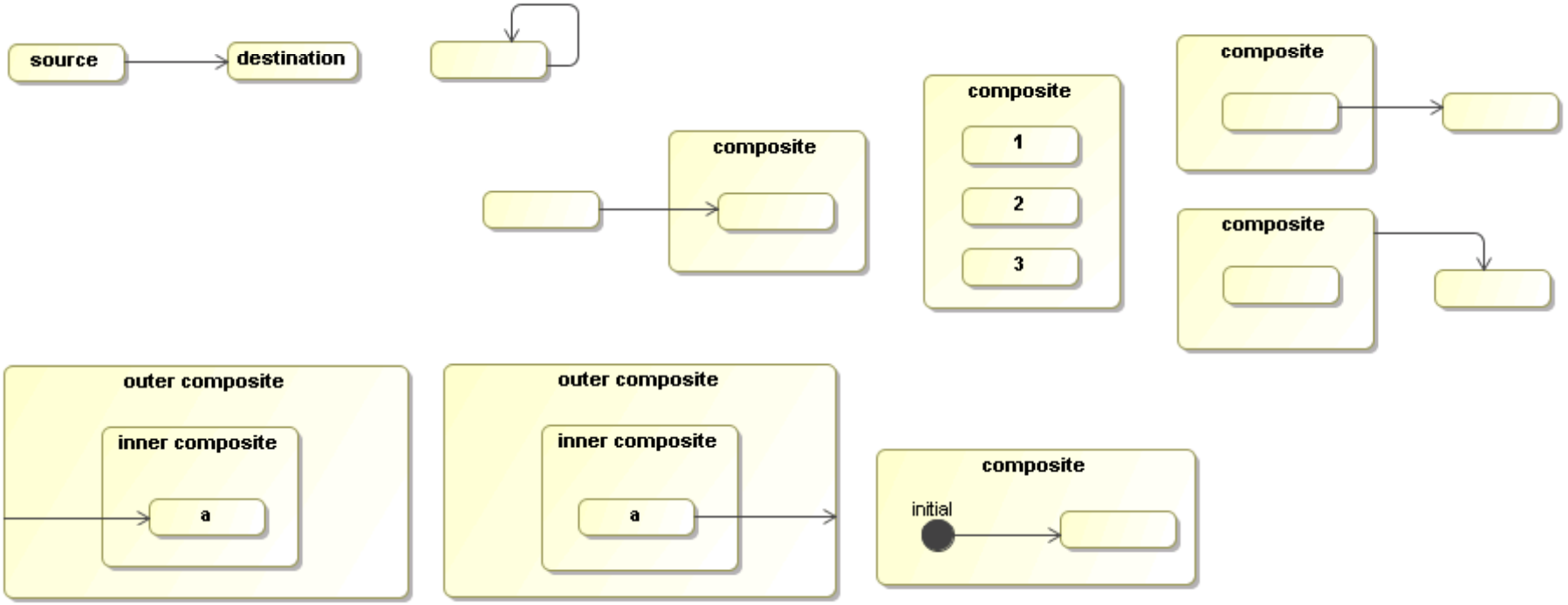


# Prozessstruktur

camunda
The Business Process Company

## Komposition ermöglicht verschiedenste Prozesskonstrukte



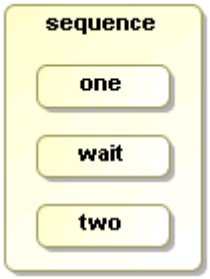



# Prozessstruktur mit Ordnung

camunda
The Business Process Company

## Sequences

```
public class Sequence implements ExternalActivity {

  public void execute(Execution execution) {
    List<Node> nodes = execution.getNode().getNodes();
    execution.execute(nodes.get(0));
  }

  public void signal(Execution execution, String signal,
                                  Map<String, Object> parameters) {
    Node previous = execution.getPreviousNode();
    List<Node> nodes = execution.getNode().getNodes();
    int previousIndex = nodes.indexOf(previous);
    int nextIndex = previousIndex+1;
    if (nextIndex < nodes.size()) {
      Node next = nodes.get(nextIndex);
      execution.execute(next);
    } else {
      execution.proceed();
    }
  }
}
```

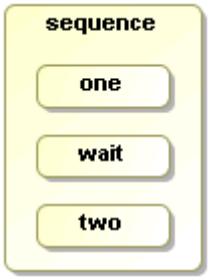


# Prozessstruktur mit Ordnung

camunda
The Business Process Company

## Sequences

```
ProcessFactory.build("sequence")
  .compositeNode("sequence").initial().behaviour(new Sequence())
    .needsPrevious()
    .node("one").behaviour(new Display("one"))
    .node("wait").behaviour(new WaitState())
    .node("two").behaviour(new Display("two"))
  .compositeEnd()
.done();
```



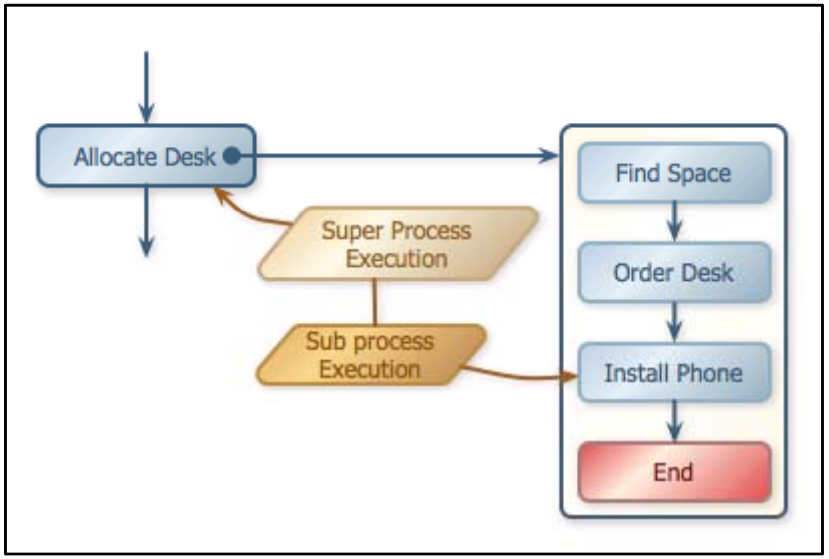

```
...
Node previous = execution.getPreviousNode();
...
```

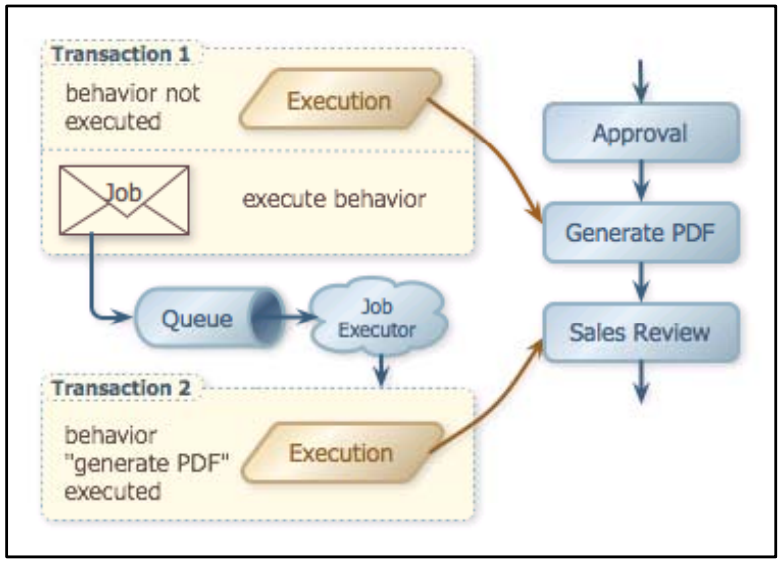
→ Vergleiche BPEL-Blöcke

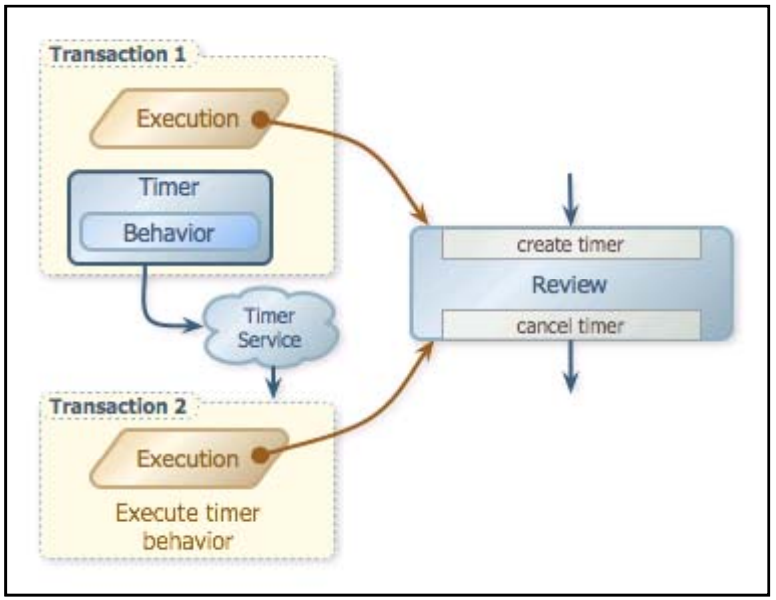
# Mehr Konzepte

camunda
The Business Process Company

## Subprozesse | Transaktionsgrenzen | Timer



# Persistenz

Viele Sprachen – eine Datenbasis

# Prozesssprachen

camunda
The Business Process Company

Die PVM unterstützt verschiedene Sprachen

- Unterstützung für Graphen & Blockstruktur
- Sprache kann durch entsprechende Nodes definiert werden
- Für Node-Typen wird Verhalten implementiert (Activities)
- Es wird geben
  - XPDL: Nova Bonita
  - jPDL: JBoss jBPM JPDL
  - BPEL
- Eigene DSL (Domain Specific Language) möglich
- MDSD-Ansätze denkbar

# JBoss jBPM jPDL

camunda
The Business Process Company

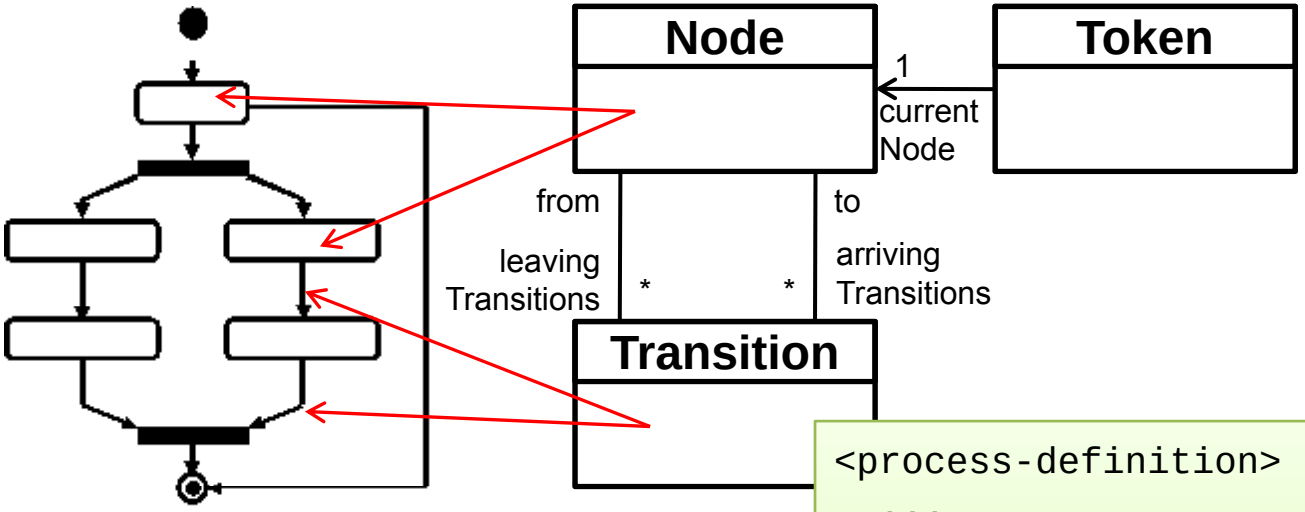
## Das Projekt

- Business Process Engine
- Proprietäre Sprache „jBPM Process Definition Language“ (jPDL)
- Aufbauend auf PVM
- Klein und flexibel, leicht erweiterbar
- Aktuell in der Entwicklung (released 3.2, noch ohne PVM), Fertigstellung bis Anfang 2009 geplant
- Java-Objekte als Prozessvariablen

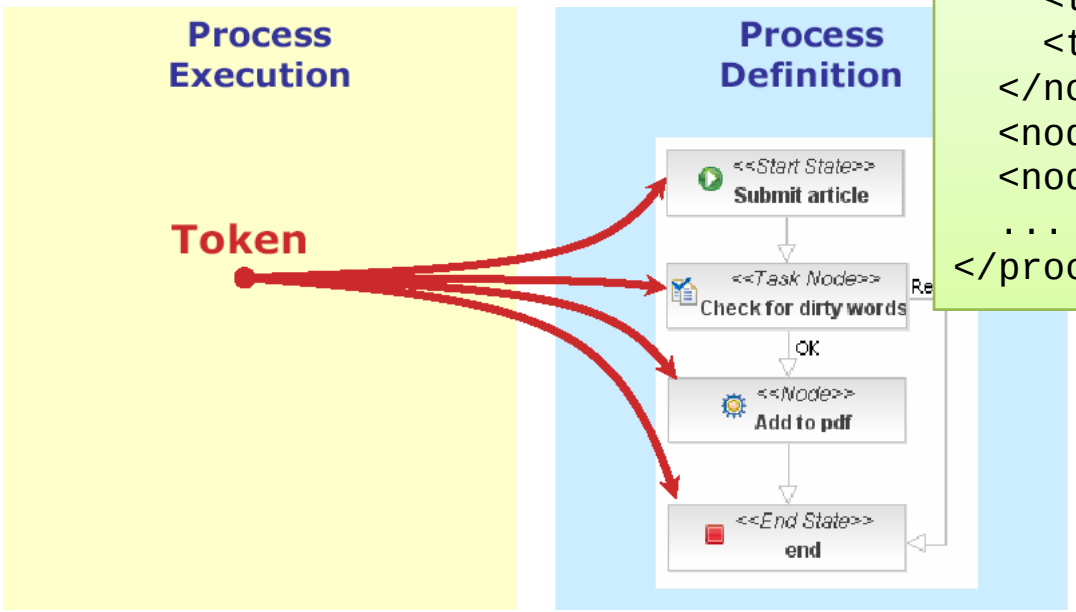
# „Graph oriented programming“ in jPDL 3

camunda
The Business Process Company

## Der Prozess als gerichteter Graph




```
<process-definition>
  ...
  <node name=“serve client”>
    <transition name=“ok” to=“order” />
    <transition name=“nok” to=“joke” />
  </node>
  <node name=“order” />
  <node name=“joke” />
  ...
</process-definition>
```

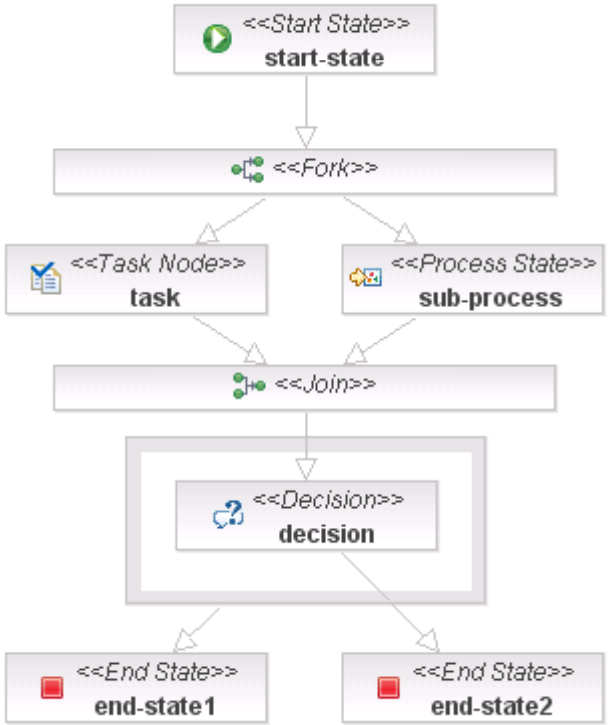


# Sprachbeispiel jPDL

camunda
The Business Process Company

## Node-Typen

- Task-Node: Human Tasks / Aufgaben
- State: Wait-States
- Fork / Join: Parallelisierung
- Decision: Automatische Entscheidung
- Start-State / End-State
- Super-State
- Process-State
- …
- Eigene Node-Typen mit Verhalten können implementiert werden



# jBPM & Java

## jBPM in a nutshell

camunda
The Business Process Company

- Einfache Java-API zur Steuerung der Engine
  - Prozessstart
  - Aufgabenliste
  - …
- Aufrufen von „User-Code“
  - definierte Stellen im Prozess
  - Interface & Java-Klassen

# jBPM & Java

camunda
The Business Process Company

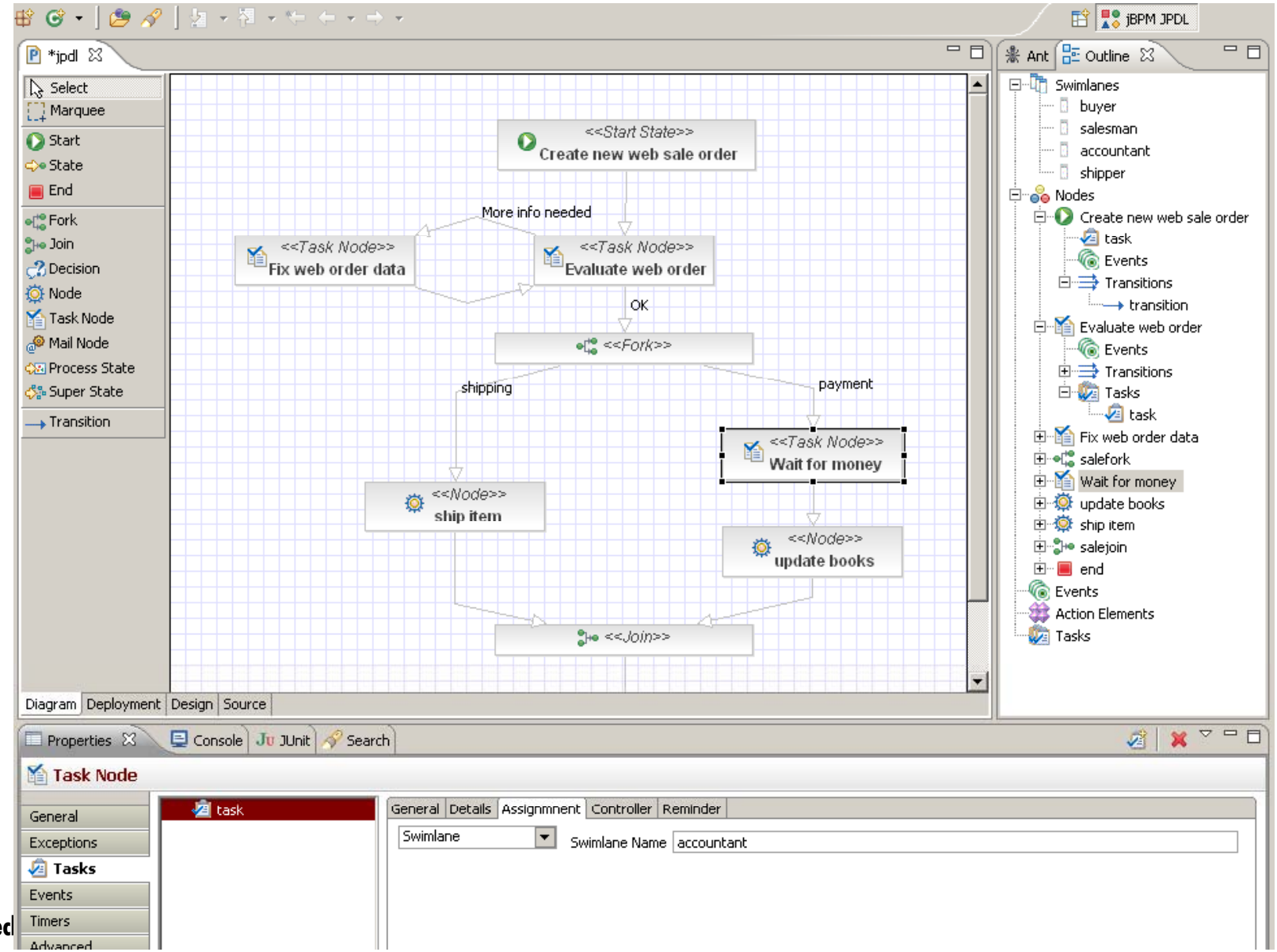
## jBPM in a nutshell

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

ProcessInstance pi = context.getGraphSession().
          findLatestProcessDefinition("Ticket").createProcessInstance();
pi.getRootToken().signal();

List<TaskInstance> tasks = context.getTaskMgmtSession().
          findTaskInstances("Vertrieb");
tasks.get(0).end("Ticket schliessen");

context.close();
```

```
public class MyAction implements ActionHandler {
 public void execute(ExecutionContext ctx) {
  Object var = ctx.getVariable("var");
  result = service.doSomething(var);
  ctx.setVariable("result", result);
 }
}
```

# Tooling: Eclipse

## Sprachbeispiel jPDL

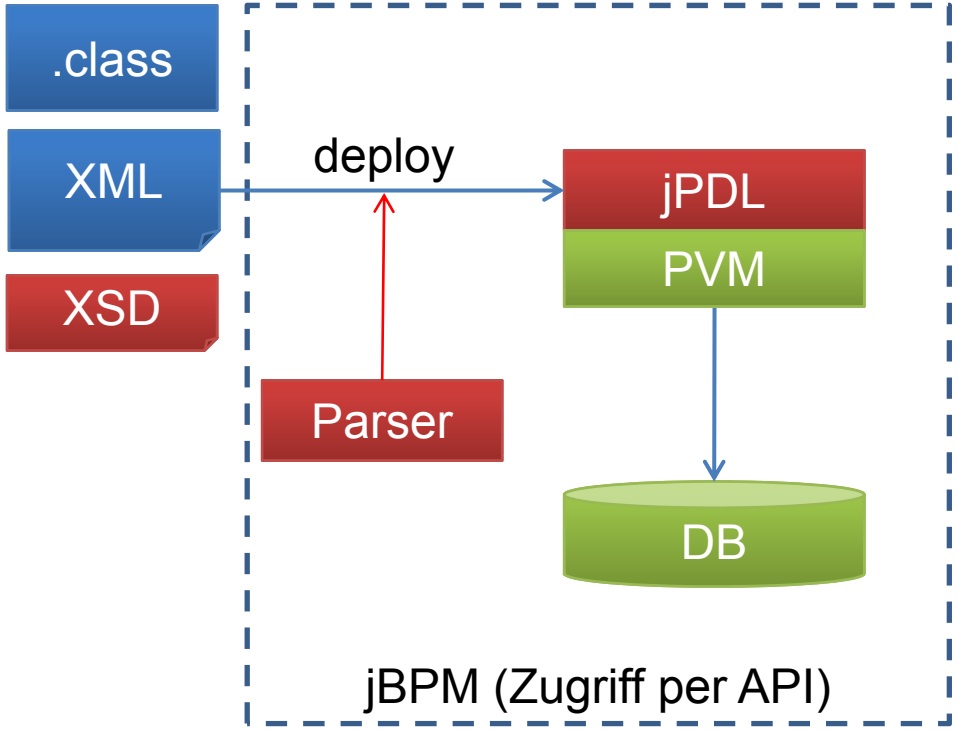


# Prozess Deployment in jBPM 4

camunda
The Business Process Company

jBPM in a nutshell

- Java API
- Ant
- Eclipse-Designer
- Web-Console
- AdminClient
- MBean
- …

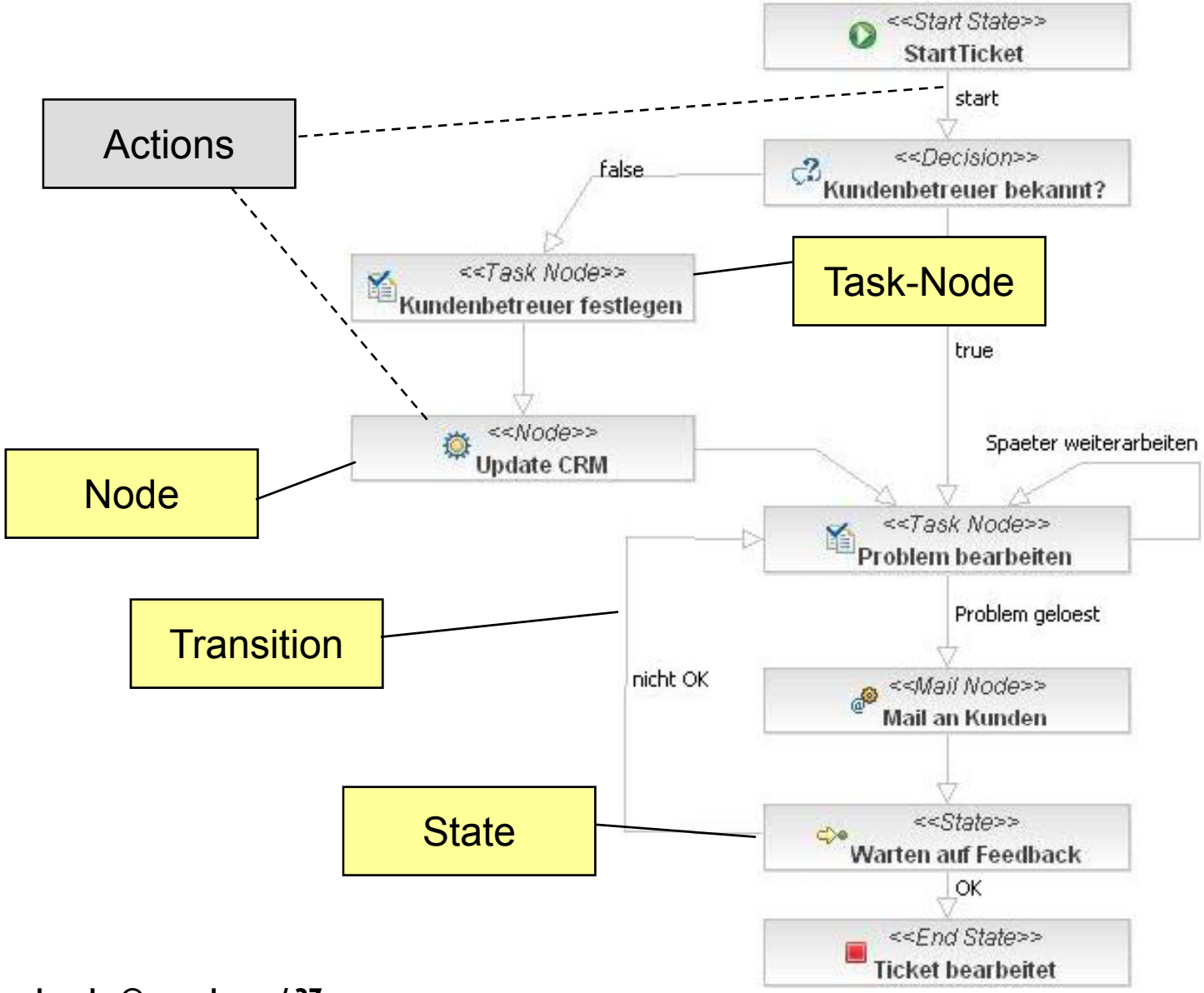



- Alle Prozesse werden versioniert
- Action-Klassen können mit deployed und versioniert werden

# Kurzer Demoprozess

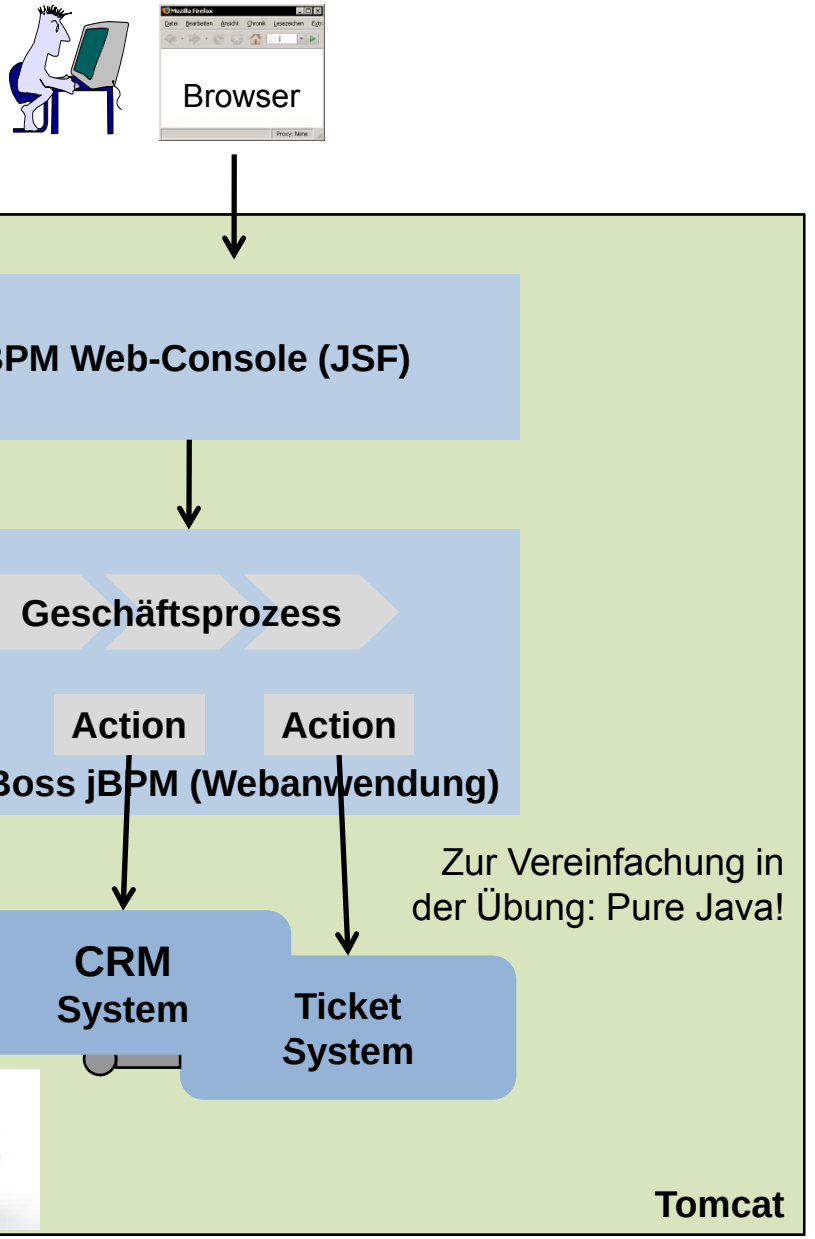
camunda
The Business Process Company

Einfacher Ticketprozess



# Demoanwendung

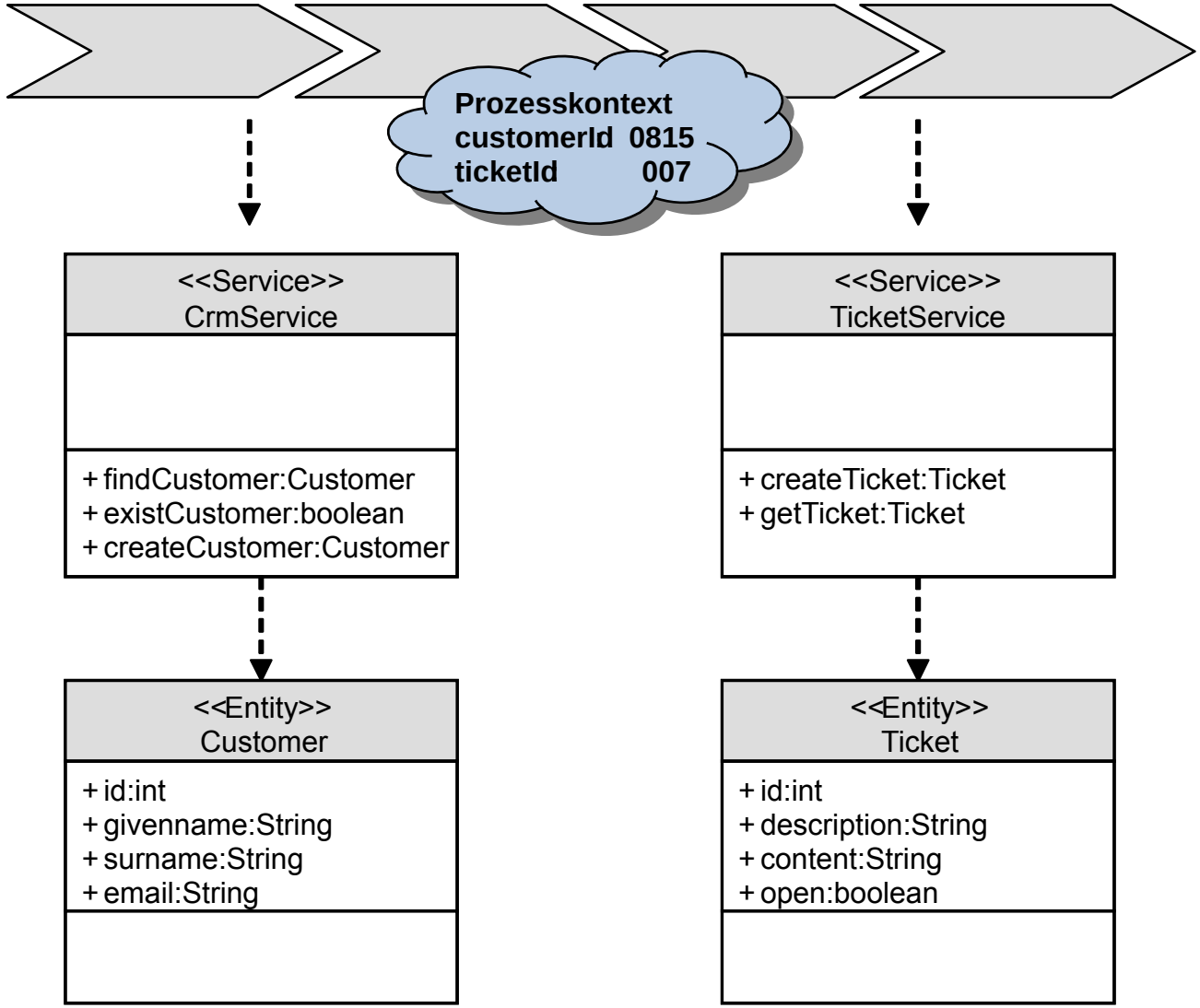
## Architektur



# CRM- & Ticketsystem

camunda
The Business Process Company

## Einfachste Java EE Implementierung & Prozessintegration



camunda
The Business Process Company

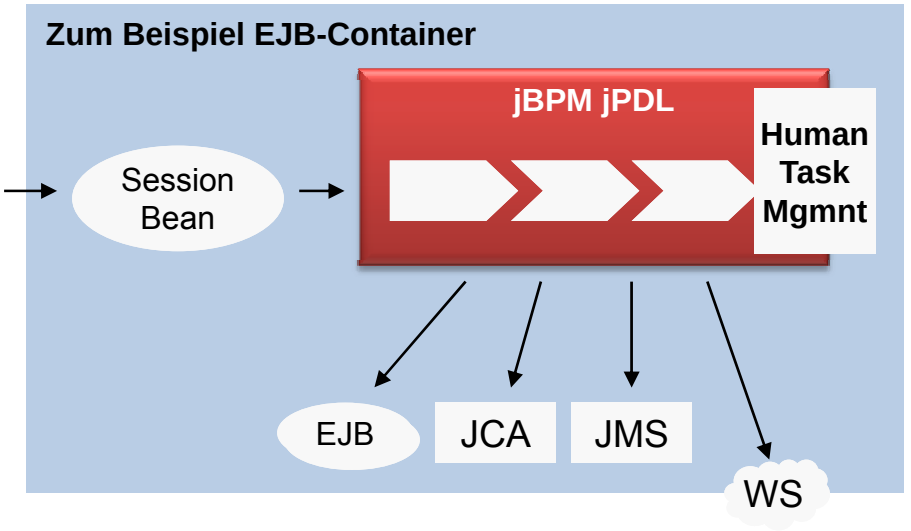
# DEMO

# jBPM in der Architektur

camunda
The Business Process Company

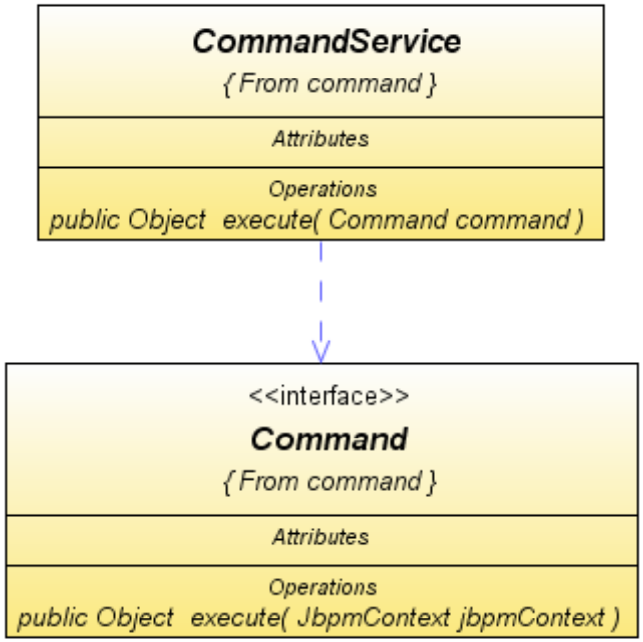
## Einbettung in eigene Anwendung

- jBPM kann in eigene Java-Architektur integriert werden
- Process Engine ist eigene Architekturschicht
- Domänenobjekte oder Referenzen als Prozessvariablen
- Ansteuerung ext. Services im Prozess



# Command-Pattern

## Remote-Zugriffe und Asynchronität

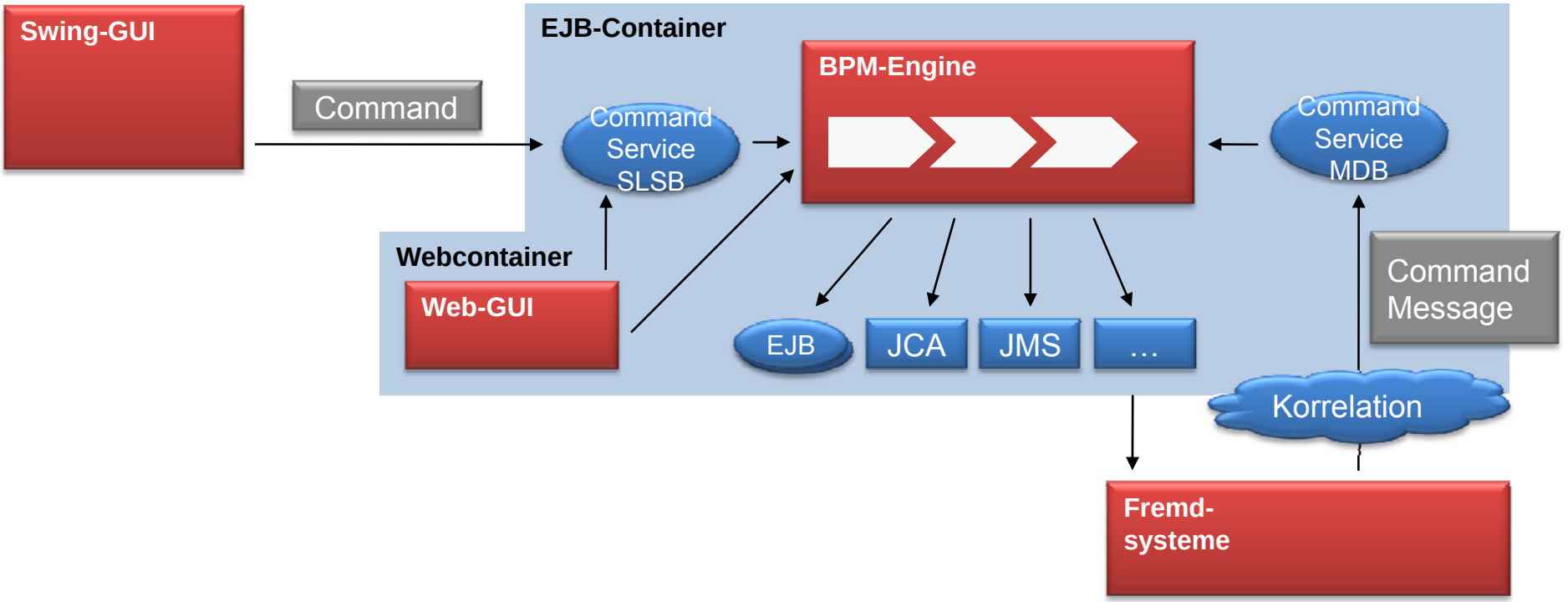



```
public class TaskInstanceEndCommand
                                  implements Command {
  ...
  public Object execute(JbpmContext jbpmContext) {
    TaskInstance taskInstance =
                        getTaskInstance(jbpmContext);

    if (transitionName == null) {
      taskInstance.end();
    } else {
      taskInstance.end(transitionName);
    }
    return taskInstance;
  }
  ...
}
```

# Beispiel: EJB3 + Swing

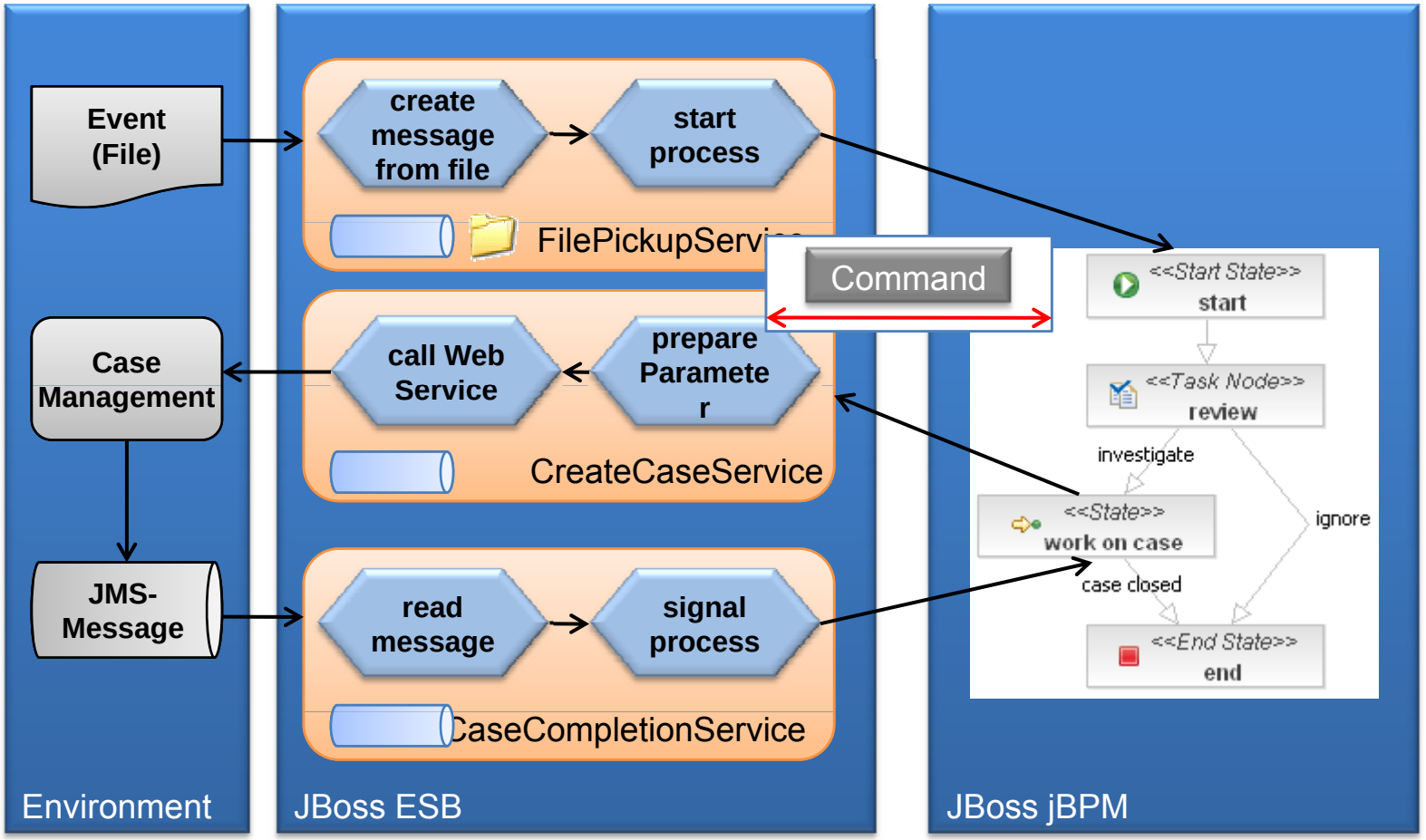
jBPM jPDL in Java Systemen

camunda
The Business Process Company



# Beispiel: JBoss ESB

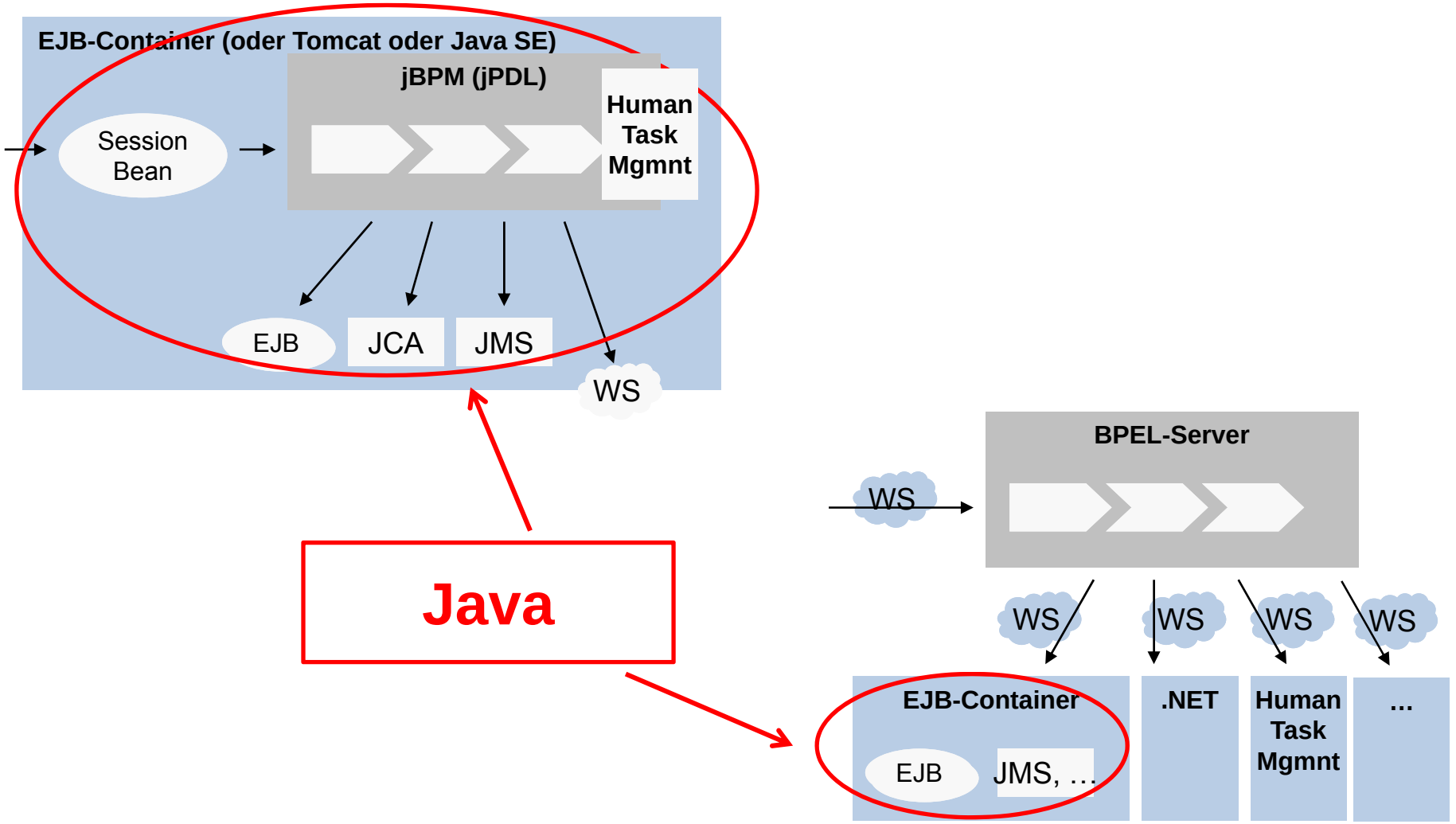
## Commands in Action



# Vergleich zu BPEL

camunda
The Business Process Company

## Java beheimatet JBoss jBPM



# Praxiserfahrungen jBPM jPDL 3

camunda
The Business Process Company

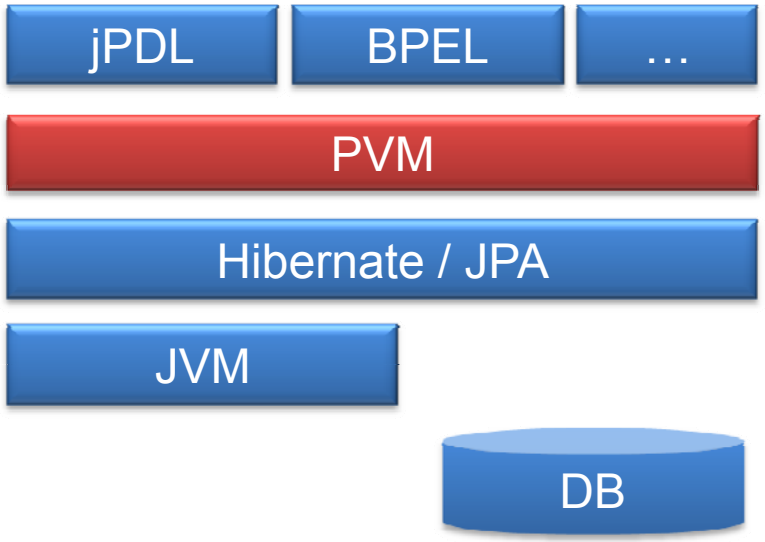
Aus dem Nähkästchen

- Performance und Skalierbarkeit bei korrektem Einsatz kein Problem
- Ohne Persistenz minimaler Overhead
- Relativ „leichtgewichtig“, kleine Lernkurve
- Gute Dokumentation für normale Probleme
- Auch in großen Projekten und Unternehmen eingesetzt

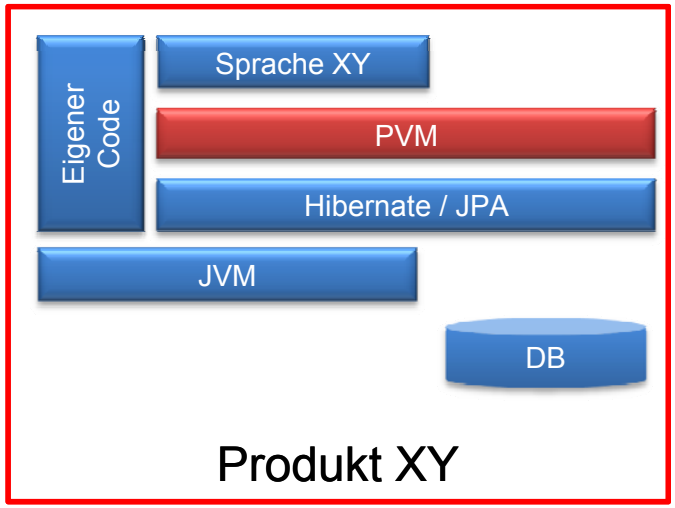
- Ach ja: Tooling (Designer und Webconsole) ist verbesserungsbedürftig ;-)

# Architektur

## PVM - Hibernate des BPM?







# Fazit & Ausblick

camunda
The Business Process Company

JBoss PVM & jBPM 4

- PVM stabilisiert sich
- Sehr interessantes Konzept, Interesse auch seitens kommerzieller Tool-Hersteller
- Im Java Umfeld ist JBoss jBPM sehr interessant
- „Leichtgewichtig“ und in verschiedensten Umgebungen Lauffähig
- Erweiterbar und flexibel
- Vision und Roadmap vorhanden!
- Einige Verbesserungen in jBPM 4: Logging konfigurierbar, Persistenz austauschbar, Control-Loop, …

# Fragen & Antworten

**Aktuelle Jobs:**
**http://www.camunda.com/jobs/**

Bernd Rücker
Geschäftsführer
Berater, Trainer & Coach
bernd.ruecker@camunda.com
+49 711 3278645

## Unsere Themen

- Ganzheitliches BPM
- Prozessautomatisierung
- SOA, BPEL, XPDL, jBPM, Drools, ESB
- BPMN
- BPM-Toolauswahl

## Unsere Leistungen

- Beratung
- Seminare
- Process as a Service (Hosting)