



jPDL: Simplified Workflow for Java

Tom Baeyens
Founder and Lead of
JBoss jBPM

It's all the same

- Workflow
- Business Process Management (BPM)
- Orchestration

...it all boils down to state machines

What is a Process Language ?

- Informal:

*Describes how
people and/or systems work together*

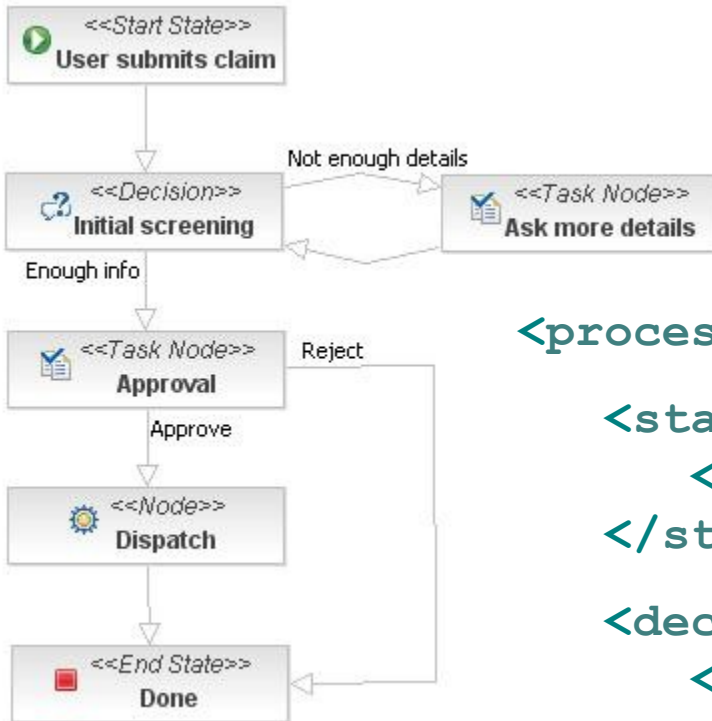
- Typical examples

✓ Insurance claim, Approvals, Legal case

- More formal:

*Describes
an **execution** flow
as **graphical** activities
that can be **wait states***

What is a Process Language ?



```
<process-definition name="Damage claim">
  <start-state name="User submits claim">
    <transition to="Initial screening"/>
  </start-state>
  <decision name="Initial screening">
    <transition name="Enough info"
      to="Approval"/>
    <transition name="Not enough details"
      to="Ask more details"/>
  </decision>
  ...
</process-definition>
```

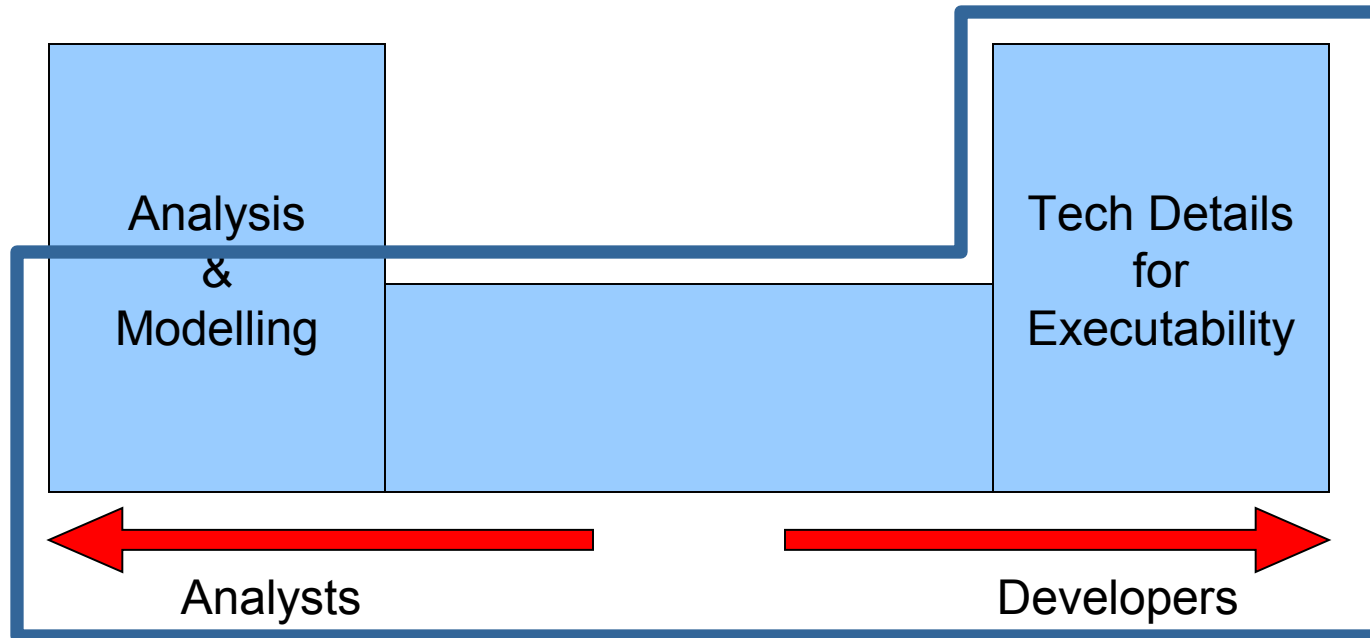
Why a Process Language ?

- Simplify implementation by extracting state management
- Improve communication
 - ✓ Modelling \leftrightarrow Execution
 - ✓ Analyst \leftrightarrow Developer
- Automatic persistent history
 - ✓ Business intelligence



What is a BPMS ?

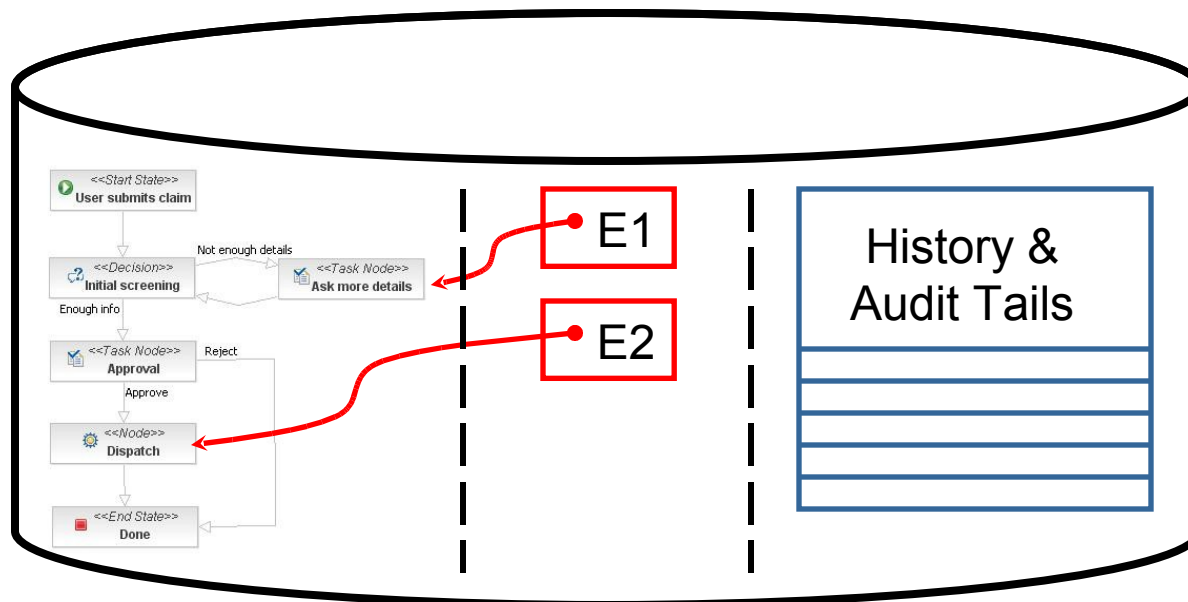
- Business Process Management System



- Modelling vs Executability
 - ✓ The free-er the modeling, the less executable

What is a BPMS ?

- Business Process Management System
- Executes processes
- Persists executions
- Records history

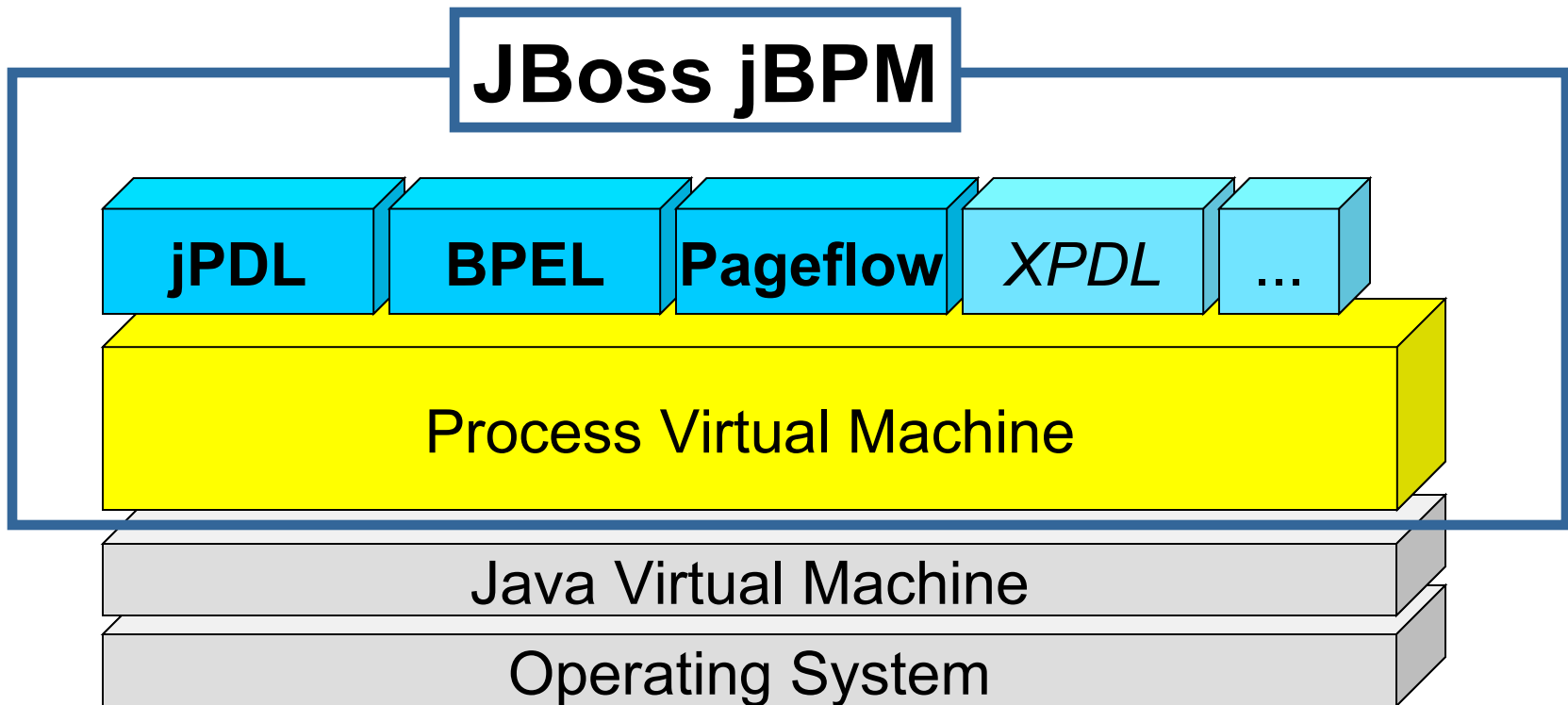


When to use a Process Language ?

- Any aspect of software development
 - ✓ Diagram
 - ✓ Some form of execution
 - ✓ Long running
- History & statistics

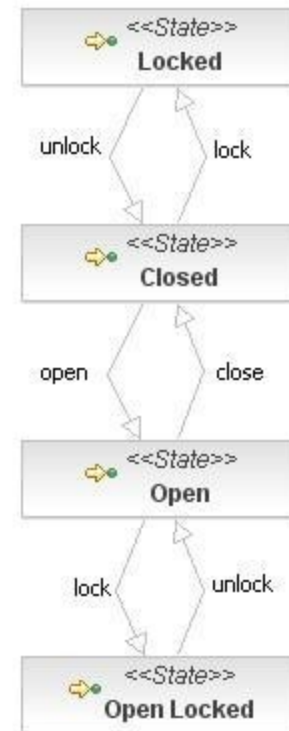
JBoss jBPM

- One embeddable technology
- Multiple process languages



A First Example

- A door
- 4 States
 - ✓ Locked, Closed, Open, Open Locked
- 4 Operations
 - ✓ unlock, lock
 - open, close
- First Java
- Then jPDL



A Door In Java

```
public class Door {
```

```
    static String OPEN = "open";
```

```
    static String CLOSED = "closed";
```

```
    static String LOCKED = "locked";
```

```
    static String OPEN_LOCKED = "open-locked";
```

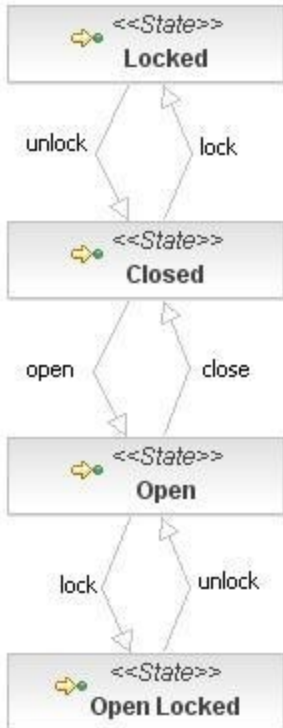
```
    String state = CLOSED;
```

```
    ...
```

A Door In Java

```
public void lock() {  
    if ( (state==LOCKED) || (state==OPEN_LOCKED) )  
    {  
        throw new IllegalStateException(  
            "door is already locked"  
        );  
    }  
  
    if (state==CLOSED) {  
        state = LOCKED;  
    } else if (state==OPEN) {  
        state = OPEN_LOCKED;  
    }  
}
```

A Door In jPDL



```
<process-definition name="door" initial="Closed">
  <state name="Locked">
    <transition name="unlock" to="Closed" />
  </state>
  <state name="Closed">
    <transition name="lock" to="Locked" />
    <transition name="open" to="Open" />
  </state>
  <state name="Open">
    <transition name="close" to="Closed" />
    <transition name="lock" to="Open Locked" />
  </state>
  <state name="Open Locked">
    <transition name="unlock" to="Open" />
  </state>
</process-definition>
```

A Door In jPDL

```
public class DoorProcessTest extends TestCase {  
  
    static ProcessDefinition doorProcess =  
        ProcessDefinition.parseXmlInputStream(  
            DoorProcessTest.class.  
getResourceAsStream("processdefinition.xml")  
        );  
  
    ...  
}
```

A Door In jPDL

token

```
public void testClosedLock() {
```

```
    ProcessInstance processInstance =
```

```
        new ProcessInstance(doorProcess);
```

```
    Token token = processInstance.getRootToken();
```

```
    token.signal("lock");
```

```
    assertEquals("Locked",
```

```
token.getNode().getName());
```

```
    token.signal("unlock");
```

```
    assertEquals("Closed",
```

```
token.getNode().getName());
```

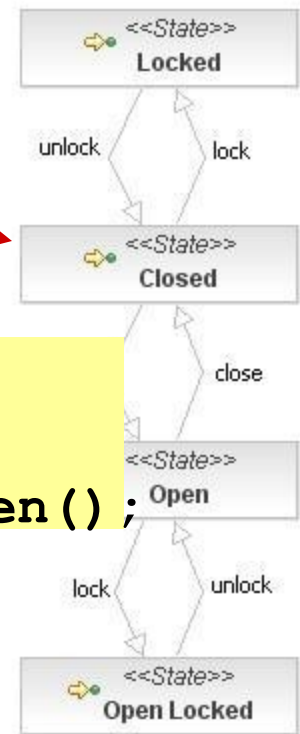
```
    try {
```

```
        token.signal("unlock");
```

```
        fail();
```

```
    } catch (JbpmException e) {
```

```
    }
```



redhat.



a division of Red Hat

A Door In jPDL

token

```
public void testClosedLock() {
```

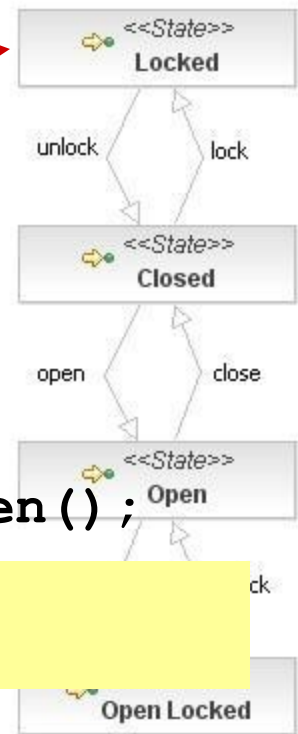
```
    ProcessInstance processInstance =  
        new ProcessInstance(doorProcess);
```

```
    Token token = processInstance.getRootToken();
```

```
    token.signal("lock");  
    assertEquals("Locked",  
token.getNode().getName());
```

```
    token.signal("unlock");  
    assertEquals("Closed",  
token.getNode().getName());
```

```
    try {  
        token.signal("unlock");  
        fail();  
    } catch (JbpmException e) {  
    }  
}
```



redhat.



a division of Red Hat

A Door In jPDL

token

```
public void testClosedLock() {
```

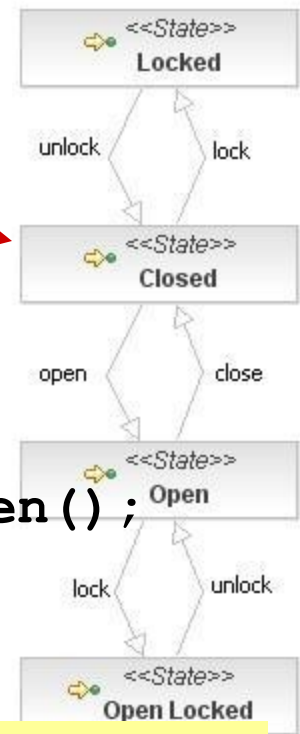
```
    ProcessInstance processInstance =  
        new ProcessInstance(doorProcess);
```

```
    Token token = processInstance.getRootToken();
```

```
    token.signal("lock");  
    assertEquals("Locked",  
token.getNode().getName());
```

```
    token.signal("unlock");  
    assertEquals("Closed",  
token.getNode().getName());
```

```
    try {  
        token.signal("unlock");  
        fail();  
    } catch (JbpmException e) {  
    }  
}
```



redhat.



a division of Red Hat

A Door In jPDL

token

```
public void testClosedLock() {
```

```
    ProcessInstance processInstance =  
        new ProcessInstance(doorProcess);
```

```
    Token token = processInstance.getRootToken();
```

```
    token.signal("lock");
```

```
    assertEquals("Locked",
```

```
token.getNode().getName());
```

```
    token.signal("unlock");
```

```
    assertEquals("Closed",
```

```
token.getNode().getName());
```

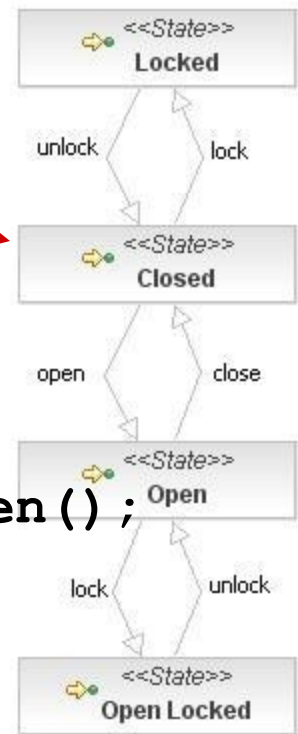
```
    try {
```

```
        token.signal("unlock");
```

```
        fail();
```

```
    } catch (JbpmException e) {
```

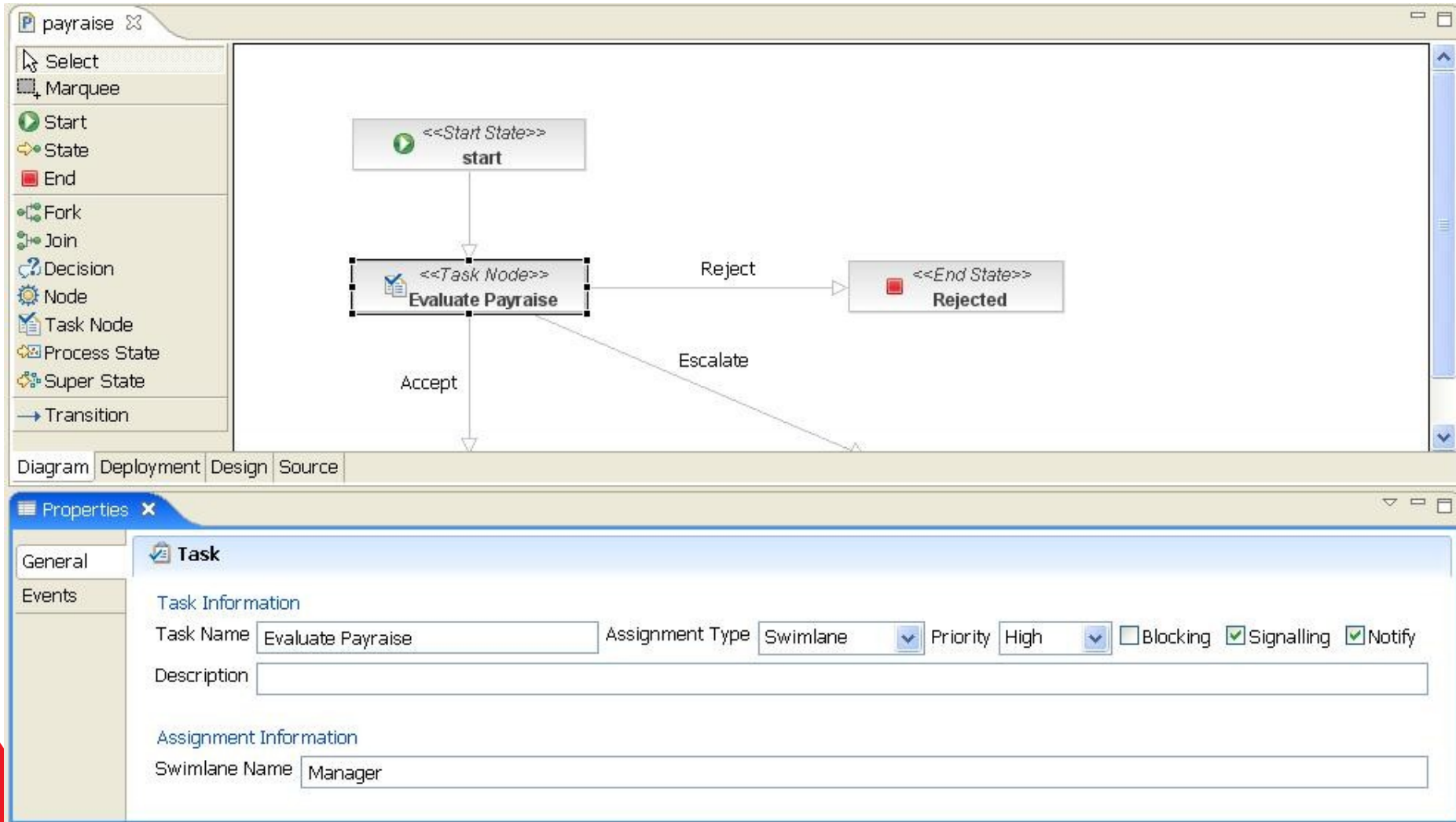
```
    }
```



redhat.



A Task In jPDL



A Task In jPDL

The screenshot shows an IDE with several tabs: PayRaiseTest.java, CalculateNumber.java, *payraise (selected), ActionHandler.java, and *Damage claim. The main editor displays jPDL code for a task. A tooltip is visible over the code, listing element types and their content models.

```
7 <start-state name="start">
8   <task name="task"></task>
9   <transition name="" to="Evaluate pay raise"></transition>
10 </start-state>
11
12 <task-node name="Evaluate pay raise">
13   <task notify="yes"
14     priority="high" swimlane="Manager">
15     <reminder duedate="2 business days"
16       repeat="4 business hours" />
17     <timer duedate="4 business days"
18       transition="Escalate" />
19   </task>
20   <transition name="Approve" to="Inform HR" />
21   <transition name="Reject" to="Notify requester" />
22   <transition name="Escalate" to="Escalate request" />
23 </task-node>
24
25
26 <>end-state
27 <>start-state
28 <t <>state
29   <>swimlane
30 <>task-node
31 # comment - xml comment
32 # XSL processing instruction - XSL processing instruction
33 </
34 </st
35
36 <state name="Update database">
37   <transition to="End" />
38 </state>
```

Element : end-state
Content Model : ()*

Diagram Swimlanes Deployment Design Source

A Task In jPDL

```
<task-node name="Evaluate pay raise">
```

```
  <task swimlane="Manager"
        notify="yes"
        priority="high">
```

```
    <reminder duedate="2 business days"
              repeat="4 business hours" />
```

```
    <timer duedate="4 business days"
           transition="Escalate" />
```

```
  </task>
```

```
  <transition name="Approve" to="Inform HR" />
```

```
  <transition name="Reject" to="Notify
requester" />
```

```
  <transition name="Escalate" to="Escalate request"
/>
```

```
</task-node>
```

Search For...

Tasks

Processes

Process Instances

Manager Menu

Deploy Process

You are logged in as:
"ernie"

Log Out

Task Instance View

Summary Task Form Source Diagram

task

Requester

Current Salary

Requested Increase

Reason

Actions

Save

Cancel

Save and Close

Task Instance Summary

Instance ID	1	Description	
Current Actor	ernie	Task Created Date	15-feb-2007 13:57:59
Pooled Actors		Task Start Date	15-feb-2007 13:58:00
Task Status	Open	Task End Date	
Task Priority	3	Task Due Date	
Process			
Instance ID	1 (View Instance)		

Actions

Reassign to

▼

Reassign

Add a comment

jPDL Is Open Ended

- Traditional BPM / workflow systems
 - ✓ **Fixed** process language
- jPDL has process constructs
 - ✓ **Base** process language, that can be extended
 - ✓ API for runtime behaviour
- jPDL can include Java code easily
 - ✓ Process == structure
 - ✓ Code as decoration

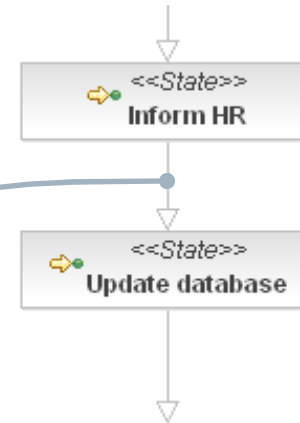
jPDL Adds Features to Java

- Use jPDL for state management
- Use the Java platform as you do now
- Natural binding



Binding Process to Code

- E.g. Action
 - ✓ Calculate number
 - ✓ On transition



```
public interface ActionHandler extends Serializable {
    void execute( ExecutionContext ctx )
        throws Exception;
}
```

Binding Process to Code

```
public class CalculateNumber
    implements ActionListener {
    String var;
    int factor;
    public void execute(ExecutionContext ctx) {
        Integer value = (Integer) ctx.getVariable(var);
        int number = factor * value.intValue();
        ctx.setVariable("number", number);
    }
}
```

Binding Process to Code

```
<state name="Inform HR">  
  <transition to="Update database">  
    <action class="payraise.CalculateNumber">  
      <var>Salary Increase</var>  
      <factor>5</factor>  
    </action>  
  </transition>  
</state>
```

Expression Language

```
<state name="Inform HR">  
  <transition to="Update database">  
    <action  
expression="#{myService.calculateNumber}" />  
  </transition>  
</state>
```

Expression Language

```
<decision name="Customer Rank ?">
```

```
  <transition to="Send expensive gift">
```

```
    <condition expression="#{customer.rank > 10}" />
```

```
  </transition>
```

```
  <transition to="Send cheap gift">
```

```
    <condition expression="#{customer.rank > 5}" />
```

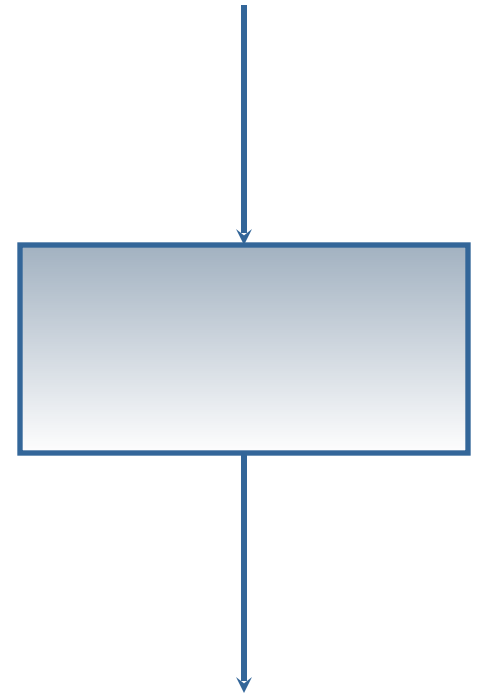
```
  </transition>
```

```
  <transition to="Check payments" />
```

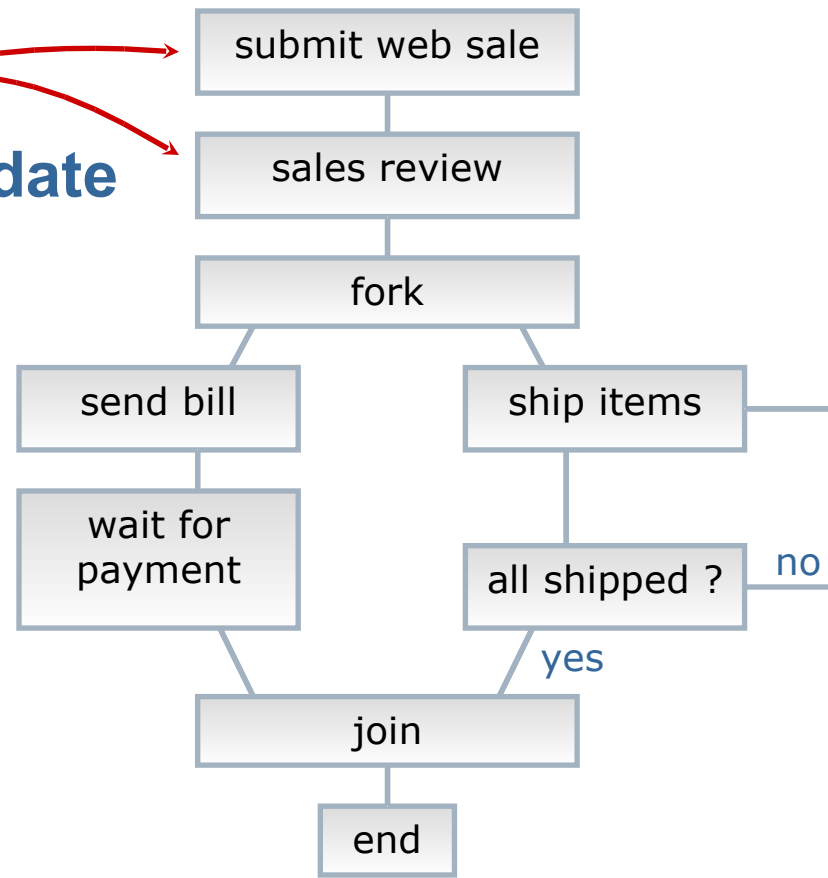
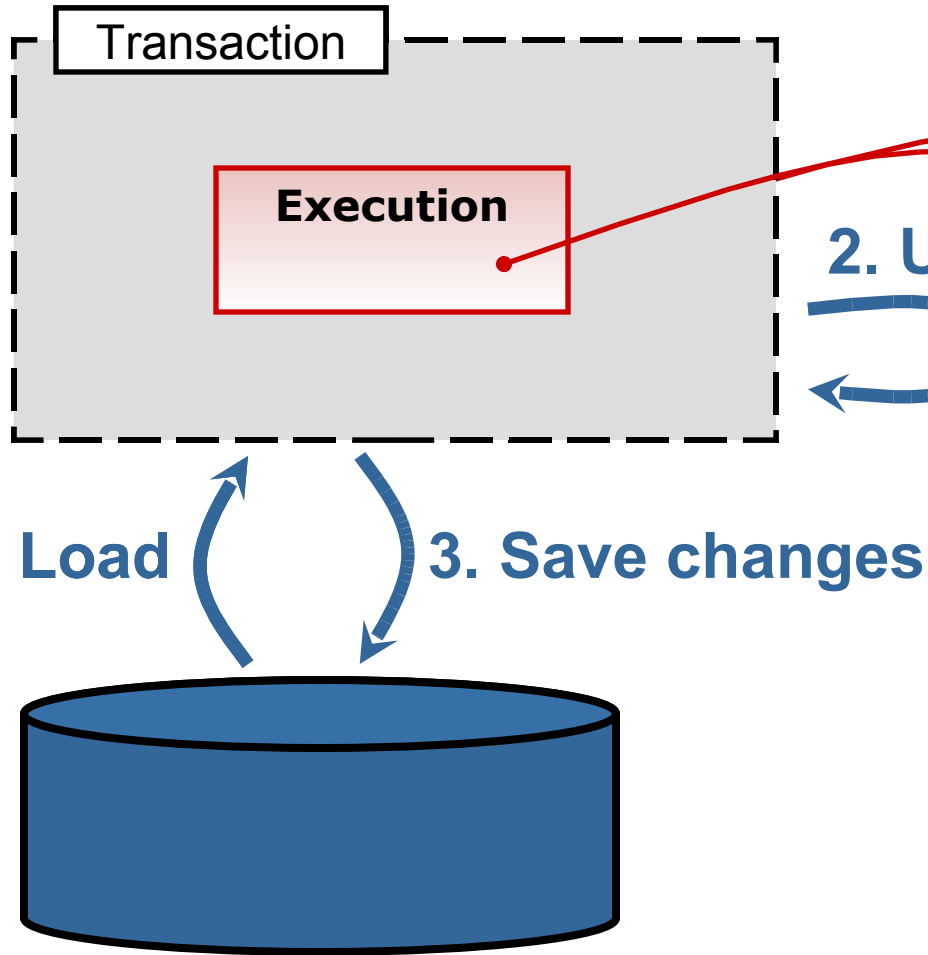
```
</decision>
```

jPDL Use Cases

- Orchestrate async architecture
 - ✓ Receiving a JMS message
 - ✓ Sending a JMS message
- Orchestrate workflow
 - ✓ Task for a person or group
- Send emails
- Invoking some java code
- Combine all of the above



jPDL Persistence



jPDL Features

- **Simple**
 - ✓ Use only what you need
- **Powerfull**
 - ✓ Superior modelling and execution
- **Embeddable**
 - ✓ Libraries
 - ✓ Database
 - ✓ Standard & Enterprise Java



Conclusion

- jPDL
 - ✓ Enriches Java
 - ✓ Extracts state management
- Simple, Powerfull, Embeddable
 - ✓ Standard & Enterprise
- JBoss jBPM
is a platform for process languages



jPDL compared to BPEL

- jPDL
 - ✓ Interface and variables are Java
 - ✓ Easy to leverage Java
 - ✓ Task management
 - ✓ Powerfull and extensible constructs
- BPEL
 - ✓ Interface and variables based on XML
 - ✓ Easy to leverage web services
 - ✓ Write a new web service as a function of others

jPDL Scalability

- Hibernate optimistic locking
 - ✓ Version column
 - ✓ `UPDATE ...`
`SET ..., version = 2`
`WHERE id = 5`
`AND version = 1`
- Synchronization by DB
- Also pessimistic locking possible
- One set of objects per transaction