redhat

JBoss®
a division of Red Hat

jPDL: Simplified Workflow for Java

Tom Baeyens
Founder and Lead of
JBoss jBPM

It’s all the same

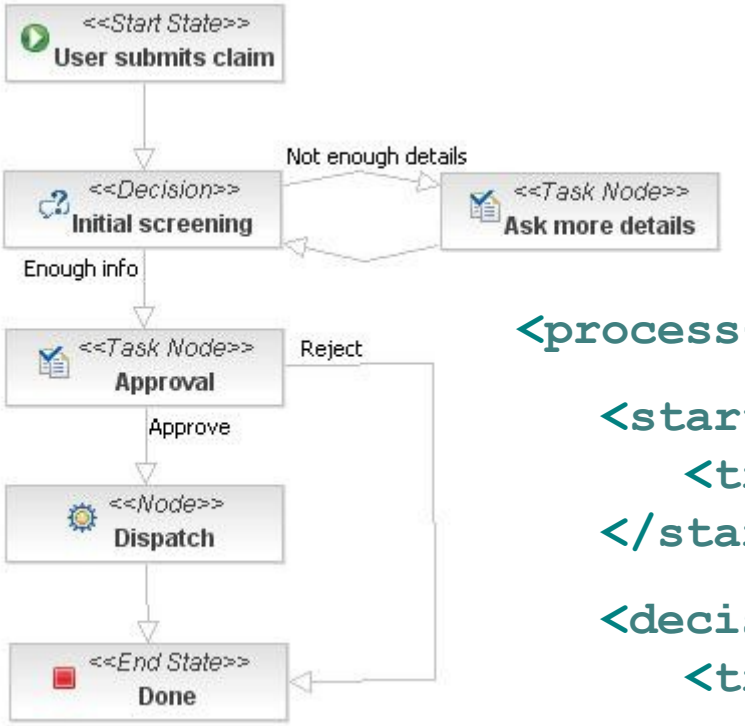
- Workflow
- Business Process Management (BPM)
- Orchestration

...it all boils down to state machines

What is a Process Language ?

- Describes how people and/or systems work together
- Typical examples
  - ✓ Insurance claim, Approvals, Legal case
- 2 targets
  - ✓ Modelling
  - ✓ Executability
- Tradeoff
  - ✓ The free-er the modeling, the less executable
  - ✓ There is some intersection

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>
   ...
```

Why use a Process Language ?

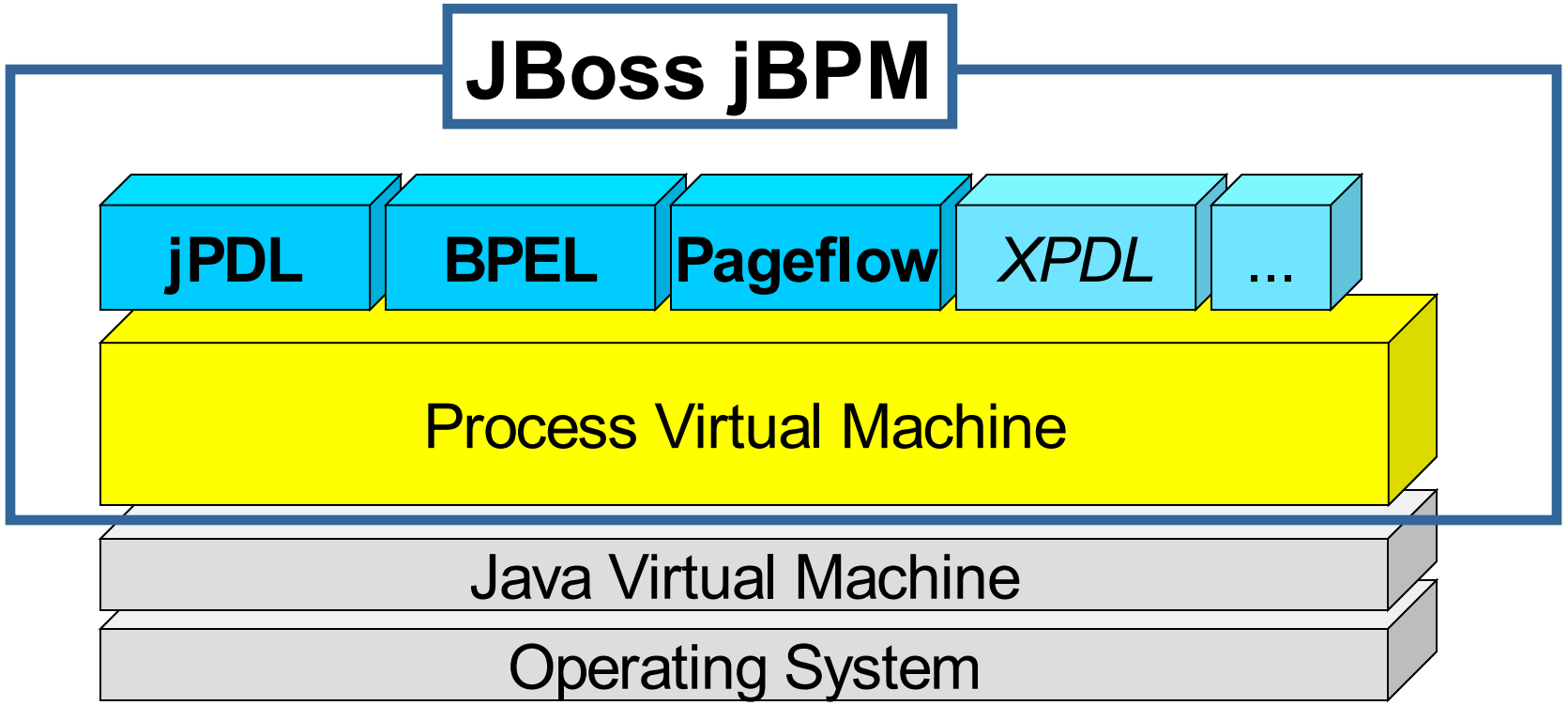
- Simplify implemention of long running processes
  - ✓ Persist process execution during wait states
- Keep the overview
- Improve communication

When to use a Process Language ?

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

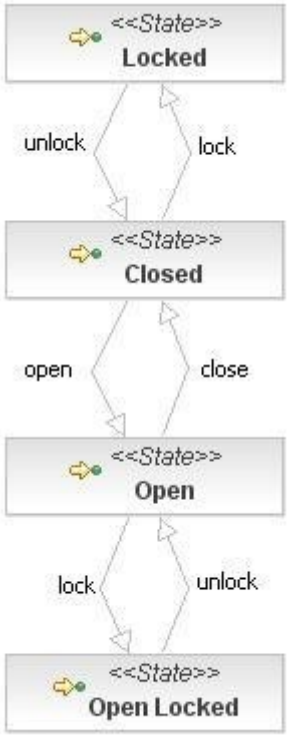
JBoss jBPM

- 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



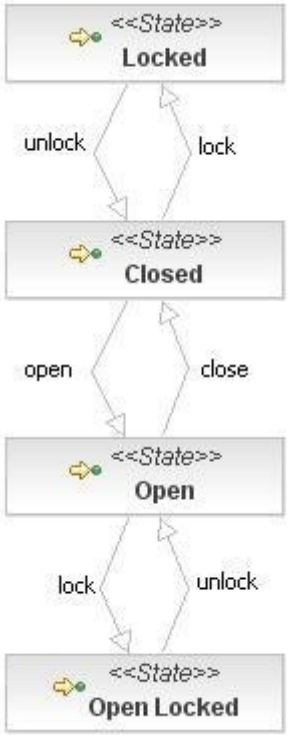
A Door In Java

```
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



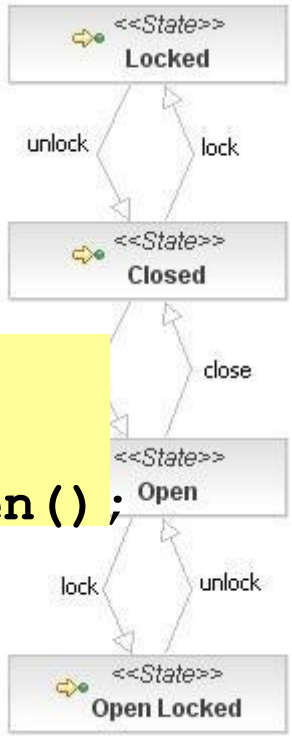

```
<process-definition name="door" initial="Closed">

  <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")
    );

  ...
```

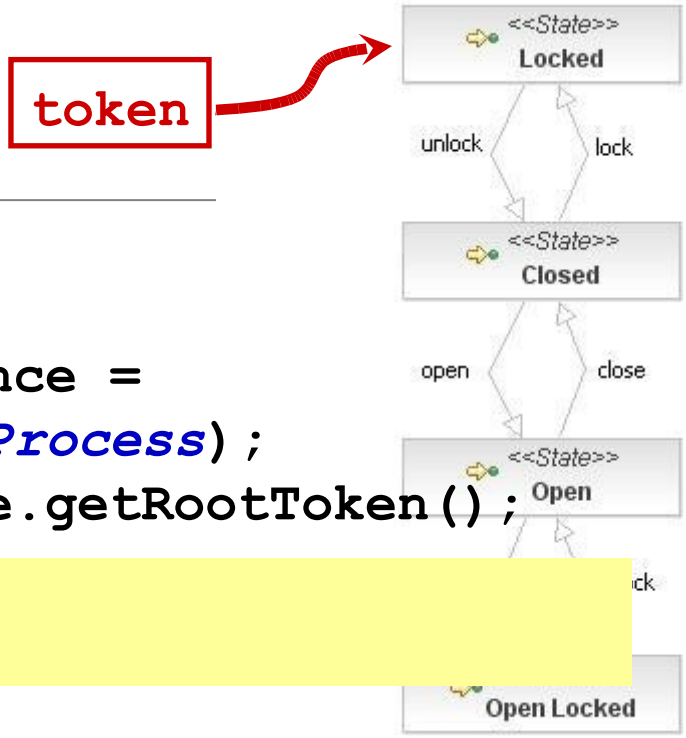
A Door In jPDL

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) {
    }
}
```

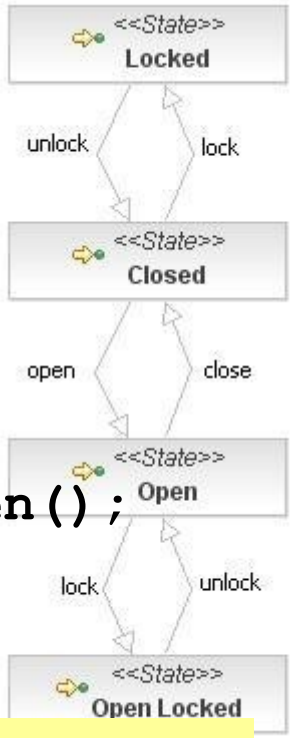
A Door In jPDL




```
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) {
  }

}
```

A Door In jPDL

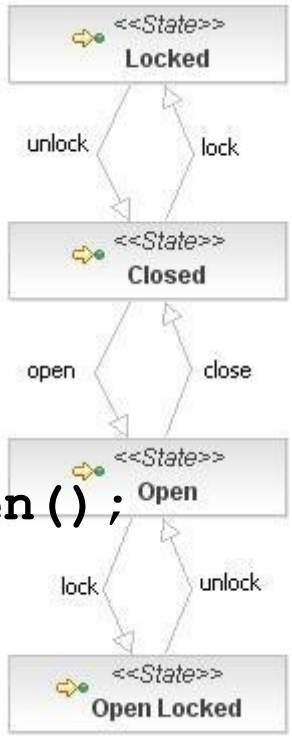
token




```
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) {
  }

}
```

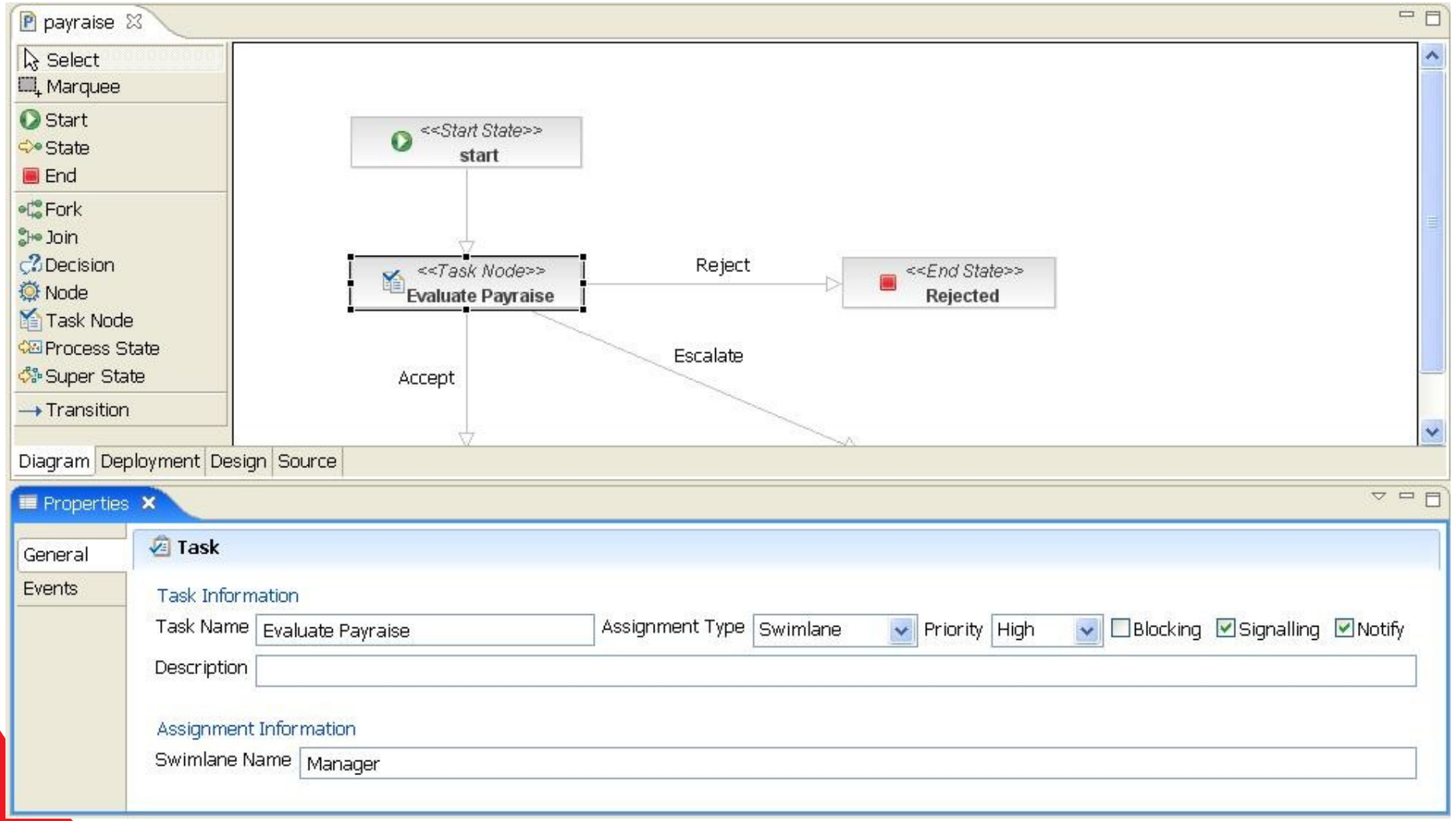
A Door In jPDL

token

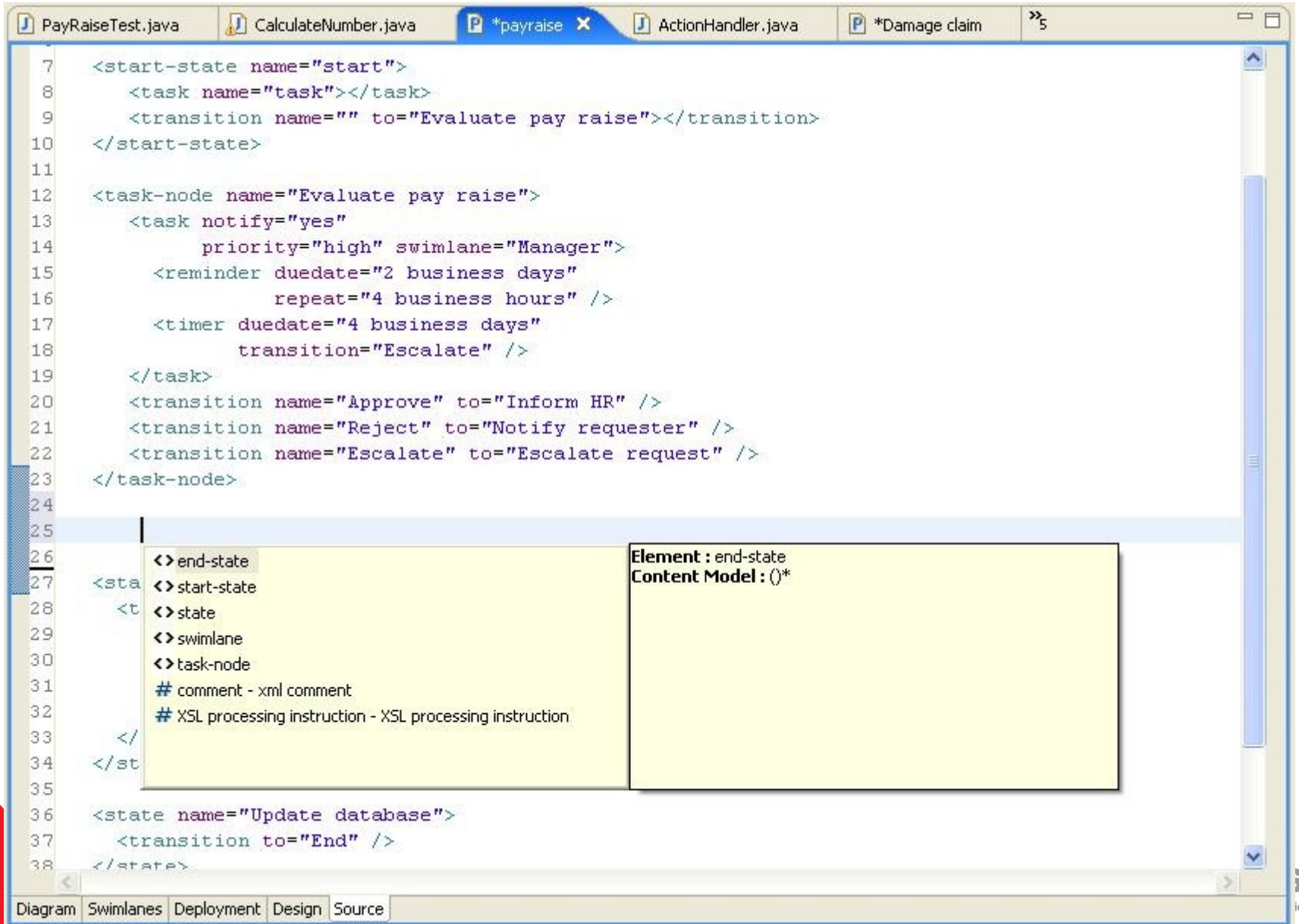



```
public void testClosedLock() {

  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) {
  }

}
```

A Task In jPDL



A Task In jPDL



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	John Doe
Current Salary	$ 81000
Requested Increase	$ 2000
Reason	I worked hard
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	cookie monster Reassign
Add a comment	

jPDL is Open Ended

- jPDL has a number of process constructs
- We saw
  - ✓ `state`
  - ✓ `task-node`
- Other node types
  - ✓ `start-state`
  - ✓ `decision`
  - ✓ `fork`
  - ✓ `join`
  - ✓ `process-state`
  - ✓ `super-state`
  - ✓ `mail-node`
  - ✓ `end-state`
  - ✓ `...`

jPDL is Open Ended

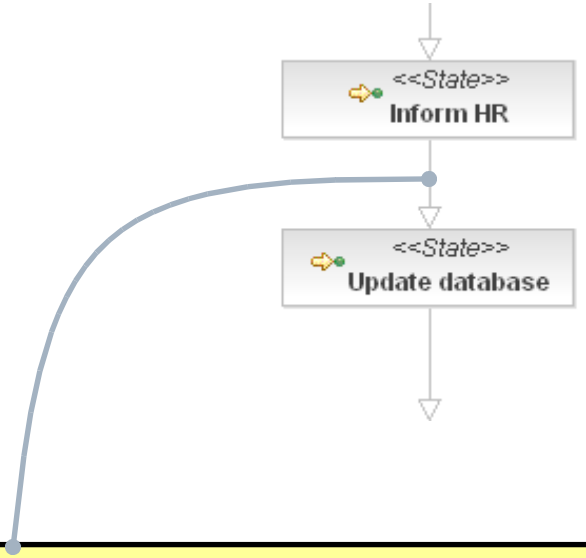
- API for runtime behaviour
  - ✓ Business logic
  - ✓ Propagation of execution
- Node types are components
- Quest for UPL is over

Binding Process to Code

- Execute piece of code in the process
- Use cases
  - ✓ Node behaviour
  - ✓ Action
  - ✓ Assignment
  - ✓ Decision
  - ✓ Timer
- Interface per use case
- Configuration in the process

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 ActionHandler {
  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="#{payraise.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 Features

- Superior modelling and execution
  - ✓ Wait states
  - ✓ Concurrent paths of execution
  - ✓ Synchronous execution
  - ✓ Asynchronous continuations
  - ✓ Clean binding to Java
  - ✓ Hidden actions
  - ✓ Timers
  - ✓ Node composition
  - ✓ Process composition

jPDL Features

- Task management
- History logging
- Process variables
- Pluggable node behaviour

jPDL Features

- Embeddable
  - ✓ Libraries
  - ✓ Database
- Pluggable services
  - ✓ Standard Java
  - ✓ Enterprise Java

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

Conclusion

- jPDL
  - ✓ Perfectly matches with Java
  - ✓ Extracts state management
  - ✓ Supports long running processes
- Embeddable
  - ✓ Standard
  - ✓ Enterprise
- JBoss jBPM
  is a platform for process languages