Declarative programming with Rules, Workflow and Event Processing

Kris Verlaenen
Core Developer
JBoss Drools
Learn how to define your business logic using a combination of both processes and rules. Why do you need it? How does it work?
A workflow engine combining processes and rules

- A workflow is a process that describes the order in which a series of steps need to be executed, using a flow chart.
Workflow Engine

- Control flow
  - Sequence, Parallelism
  - Choice, Loop, ForEach
- Data flow
- Nodes
  - Action, Event Wait (State), Subflow, Timer, Fault, Event, Human Task, Work Item
Drools Flow

A workflow engine combining processes and rules

- Integration
  - From loose coupling (decision services)
  - To advance integration (process rules)
- Unification
  - Rules and processes are different types of business knowledge assets
  - API + Tooling (IDE, repository, management, etc.)
Decision Service

- RuleSet node
- Ruleflow-group

Advantages
- Loosely coupled
- Different scope and life cycle
- No data passing

```
rule "High Risk if age < 21"
  ruleflow-group "RiskAssessment"
  when
    Person( age < 21 )
  then
    insert ( new RiskFactor(
        0.1, "Person is less than 21." ) );
```

```
end
```
Process Rules

• Rule conditions can be used to express constraints in a process
  • Choice, event wait, etc.

• Advantages
  • Tightly coupled
  • Complex conditions
  • Data-centric
  • Performance
  • Declarative, higher-level
Rules and Processes

- Process Rules
- Decision Services

SCOPE

COUPLING

generic

process-specific

tightly coupled

loosely coupled
Why integrate rules and processes?

- Simplicity
- Agility
- Granularity
- Declarativeness
- Scope
- Data-centric
- Performance
- High-level
- Unification
Simplicity

Some business logic might be easier to specify using a set of rules.
Agility

Rules and processes can have a separate life cycle

```
rule Decision1
  when
  // conditions
  then
  // actions
  end
```
Granularity

Rules can handle specific circumstances, processes are more about overall control flow

90%

5%

3%

2%
Why integrate rules and processes?

- Simplicity
- Agility
- Granularity
- Declarativeness
- Scope

- Data-centric
- Performance
- High-level
- Unification
How does it work?

- Business decisions are hard-coded inside the processes

- Better: Extract business decision using rules
Example

- Business decisions are externalized using a decision service

```
rule Decision1
when
  // conditions
then
  // actions
end
```
Example

• What if there is a lot of business logic like this?
Flow of Control

Process Engine

Rules Engine
Inversion of Control
Integration Examples

- RuleSet: Evaluating a set of rules in your process (~ decision service)
- Using rules for evaluating constraints (= process rules)
- Assignment rules
- Describing exceptional situations using rules
- Modularizing concerns using rules
- Using rules to dynamically alter the behavior of the process
- ...
  
www.devoxx.com
Domain-specific Processes

Start → Finder → For Each File
- Log
- Check File

Archive → Copy → Email → End

Recipients:
<table>
<thead>
<tr>
<th>Type</th>
<th>Display Name</th>
<th>E-Mail</th>
</tr>
</thead>
<tbody>
<tr>
<td>to</td>
<td>Tom Schindl</td>
<td><a href="mailto:tom.schindl@bestsolution.at">tom.schindl@bestsolution.at</a></td>
</tr>
<tr>
<td>bcc</td>
<td>Boris Bokowsdi</td>
<td><a href="mailto:boris_bokowsdi@ca.ibm.com">boris_bokowsdi@ca.ibm.com</a></td>
</tr>
<tr>
<td>cc</td>
<td>Tod Creasey</td>
<td><a href="mailto:tod_creamsey@ca.ibm.com">tod_creamsey@ca.ibm.com</a></td>
</tr>
</tbody>
</table>

New Application

General   Attachments   Connection   Error Handling

From: 
To: 
Reply To: 
Subject: 
Type: [text] [html]  
Body: 

[Diagram showing flow with components and work items]
Domain-specific process example
Domain-specific processes
Drools Flow

Unifies rules and processes in a single engine

- Ability to use rules everywhere in your process
  - Decision nodes, constraints, exception and event handling, task assignment, etc.
- Processes and rules see, reason and react on the same data
  - No data passing or synchronization
- Processes and rules interact
- Integrated API + tooling
Knowledge-based API

```java
KnowledgeBuilder kbuilder =
    KnowledgeBuilderFactory.newKnowledgeBuilder();

kbuilder.add(ResourceFactory
    .newClassPathResource("rules.drl"), KnowledgeType.DRL);

kbuilder.add(ResourceFactory
    .newClassPathResource("process.rf"), KnowledgeType.DRF);

KnowledgeBase kbase = KnowledgeBaseFactory.newKnowledgeBase();

kbase.addKnowledgePackages(kbuilder.getKnowledgePackages());

StatefulKnowledgeSession ksession = kbase.newStatefulKnowledgeSession();

ksession.insert(new Message("Hello"));
ksession.fireAllRules();

ksession.startProcess("com.sample.ruleflow");
```
Integrating debug & audit
Additional Features

• Extensible process framework
• Reusable set of core nodes
• Persistence of process instances
• Binary serialization of process instance, JPA
• XML format, Factory API
• Domain-specific work items
• Integrated debug and audit
• Process skins
<process xmlns="http://drools.org/drools-4.0/process" type="RuleFlow" name="ruleflow"
  id="com.sample.ruleflow" package-name="com.sample">
  
  <nodes>
    <start id="1" name="Start" />
    <actionNode id="2" name="Hello">
      <action type="expression" dialect="mvel">
        System.out.println("Hello World");
      </action>
    </actionNode>
    <end id="3" name="End" />
  </nodes>

  <connections>
    <connection from="1" to="2" />
    <connection from="2" to="3" />
  </connections>

</process>
RuleFlow Process Factory API

RuleFlowProcessFactory factory =
    RuleFlowProcessFactory.createProcess(
        "com.sample.ruleflow");

factory
// header
    .name("ruleflow").packageName("com.sample")
// nodes
    .startNode(1).name("Start").done()
    .actionNode(2).name("Hello")
    .action("java",
        "System.out.println("Hello World")");
    .done()
    .endNode(3).name("End").done()
// connections
    .connection(1, 2)
    .connection(2, 3);
RuleFlowProcess process =
    factory.validate().getProcess();
Rules, Process and Event Integration Example

Shows how rules, processes and events could be combined using some simple fictitious application example.
Case Study

**CompuSales**

- Fictitious company selling computers and related items on the internet
- Step-by-step example on how to use rules, processes and events to define and manage your business logic
Roadmap

- **Step 1**: Extract business logic for discount and validation using rules

- **Step 2**: Create a process that defines the normal flow when handling orders

- **Step 3**: Use event processing to
  - dynamically adapt business logic
  - monitor sales
Domain Model

Order
- orderId
- customerId
- date
- discount
- orderItems

Customer
- customerId
- firstName
- lastName
- birthDate

Order Item
- itemId
- amount
- price

Item
- itemId
- name
- description
- minimalAge
Overview

End User → Internet → Application Server → Database

Web Application → Services
Step 1: Rules

End User → Internet → Application Server → Database

Web Application → Services → Drools Engine → Knowledge Repository

Validation Rules
Discount Rules
Step 1a: Validation Rules

rule "Minimal age"
  when
    o: Order() |
    c: Customer() from
      customerService.getCustomer(o.getCustomerId()) |
    oi: Order.OrderItem() from o.getOrderItems() |
    i: Item(minimalAge > (c.getAge())) |
      from itemCatalog.getItem(oi.getItemId()) |
  then
    System.err.println("Minimal age violated!");
    o.addError("Minimal age violated ") +
Step 1b: Discount Rules

rule "5% discount after 18h"
  when
    There is an Order
    - with order date after 18 hours
  then
    Set discount percentage to 5 %
end
Step 1c: Knowledge Repository
Step 2: Processes

End User → Internet → Application Server → Database

- Web Application
- Services
- Drools Engine
- Order Process
- Validation Rules
- Knowledge Repository
Step 2: Process
Process and rules example
Step 3: Events

End User → Internet → Application Server → Database

Web Application → Services

Drools Engine

CEP Rules

Validation Rules

Knowledge Repository
Events

• Dynamically adapt business logic
  • Add additional logging when a problem has been detected.
    • Dynamically deploy rule that adds logging information
  • Dynamically apply discounts if sales are low
    • CEP-based discount rule
• Monitor business processes
  • Generate alerts
Step 3: Events

• Monitor sales

• Use events generated by the engine (when executing rules and processes) to monitor your business

• User-defined charts show key business indicators
  • Eclipse BIRT

• Use CEP rules to
  • Derive higher-level events
  • Generate alerts
Business Activity Monitoring

Overview of current order status

Total Amount per Day

Orders per Day

Customers per Day

www.devoxx.com
Items

Outstanding Orders
rule "Number of process instances above threshold"

    when

        Number( nbProcesses : intValue > 10 )

    from accumulate(
        e: ProcessStartedEvent( ) over window:size(1h),
        count(e) )

    then

        System.err.println( "WARNING: Number of order
        processes in the last hour above 10: " +
        nbProcesses );

    end
Process, rules and events example
Roadmap

• Pluggable and reusable composite nodes
  • Work item repository
  • Guvnor integration for installation / management
• Adaptive systems / extend power of IoC
• Advanced remoting capabilities
• Flow Editing in Guvnor
• General BPM / BAM
• Human task management
  • WS-HT + minimal Eclipse/guvnor
• Extend workflow pattern support, more advanced node types
Thanks for your attention!

Drools Homepage
http://www.jboss.org/drools/