

Declarative programming with Rules, Workflow and Event Processing



Kris Verlaenen
Core Developer
JBoss Drools

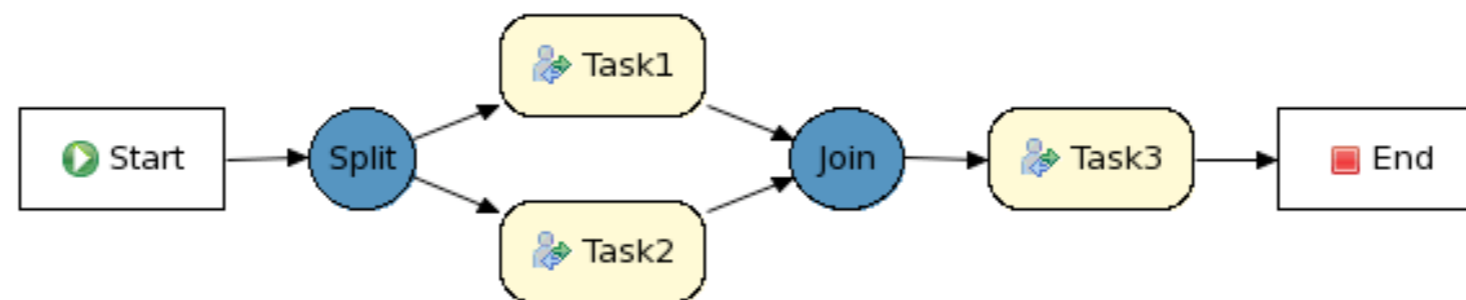
Process and Rules Integration

Learn how to define your business logic using a combination of both processes and rules. Why do you need it? How does it work?

Drools Flow

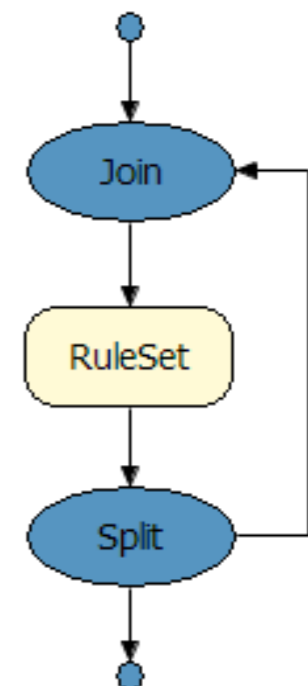
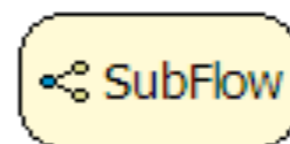
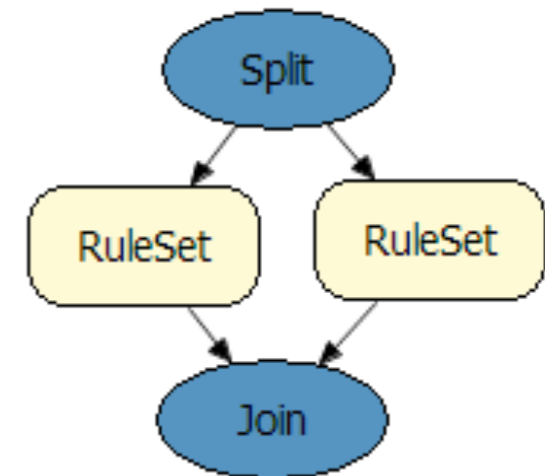
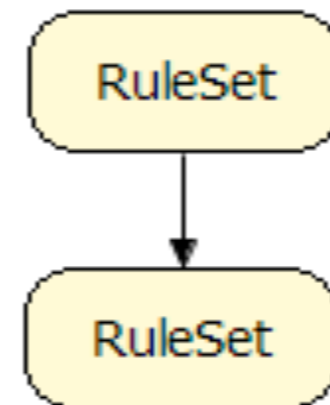
A workflow engine combining processes and rules

- 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, Paralellism
 - Choice, Loop, ForEach
- Data flow
- Nodes
 - Action, Event Wait (State), Subflow, Timer, Fault, Event, Human Task, Work Item



Drools Flow

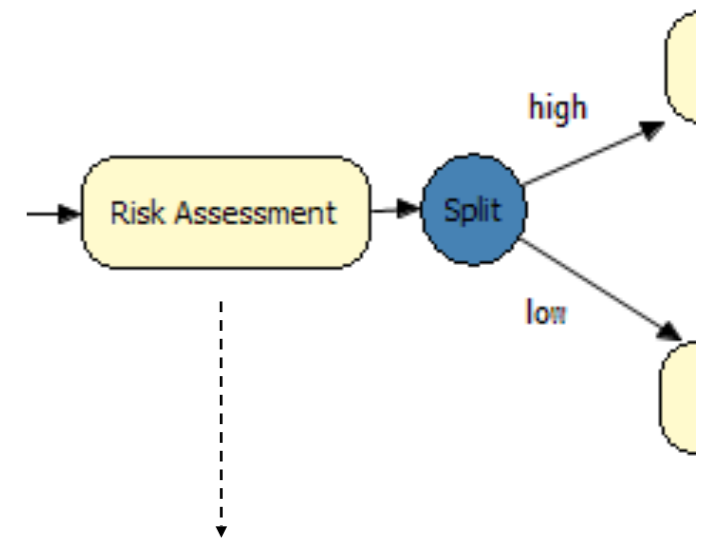
A workflow engine combining processes and rules

- 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

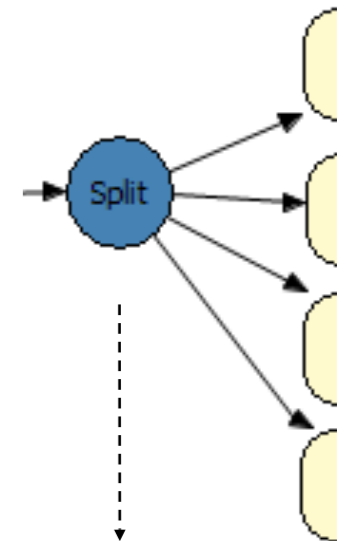
- RuleSet node
 - Ruleflow-group
- Advantages
 - Loosely coupled
 - Different scope and life cycle
 - No data passing



```
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  
end  
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



The screenshot shows a 'Constraint editor' dialog box with the following fields and controls:

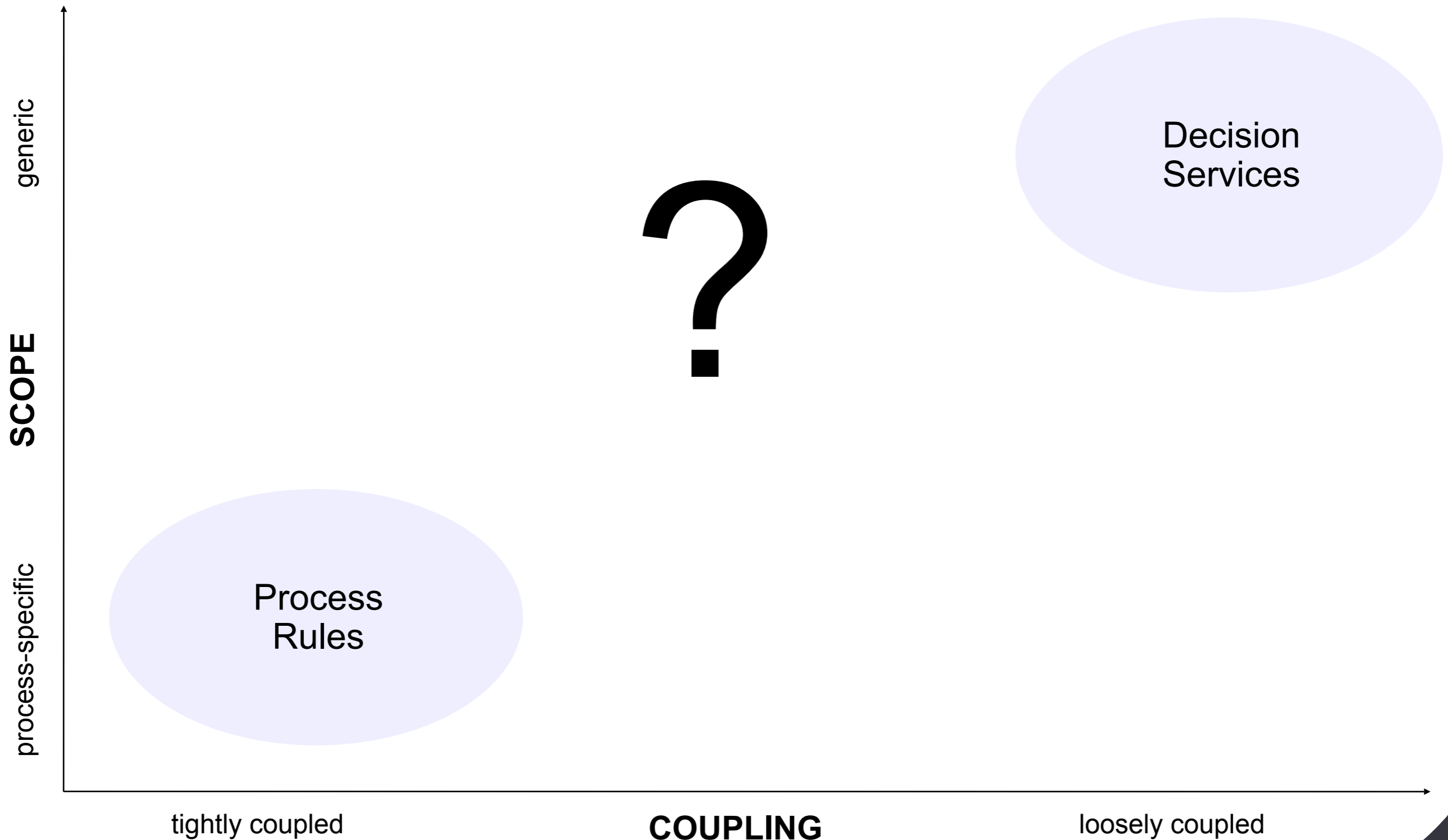
- Name: High Risk
- Priority: 1
- Always true
- Imports ...
- Globals ...
- Type: rule
- Dialect: mvel

The Textual Editor contains the following text:

```
p: Person ( age > 18 and < 60 )
not RiskEval ( risk > 0.8 ) from history.getRiskEvaluations(p.id)
Account ( amount > 10.000$ ) from p.getAccounts()
```

Buttons: OK, Cancel

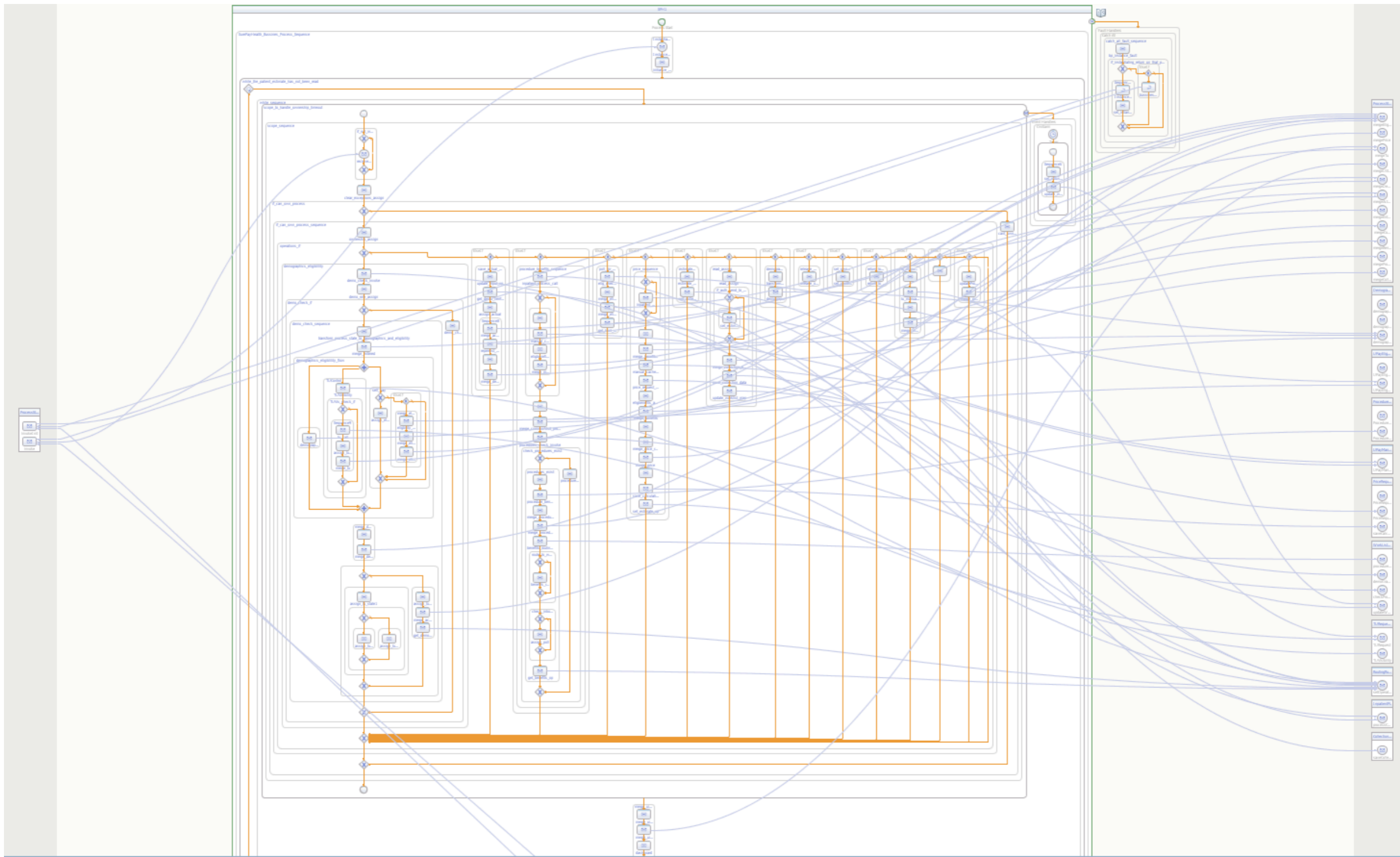
Rules and Processes



Why integrate rules and processes ?

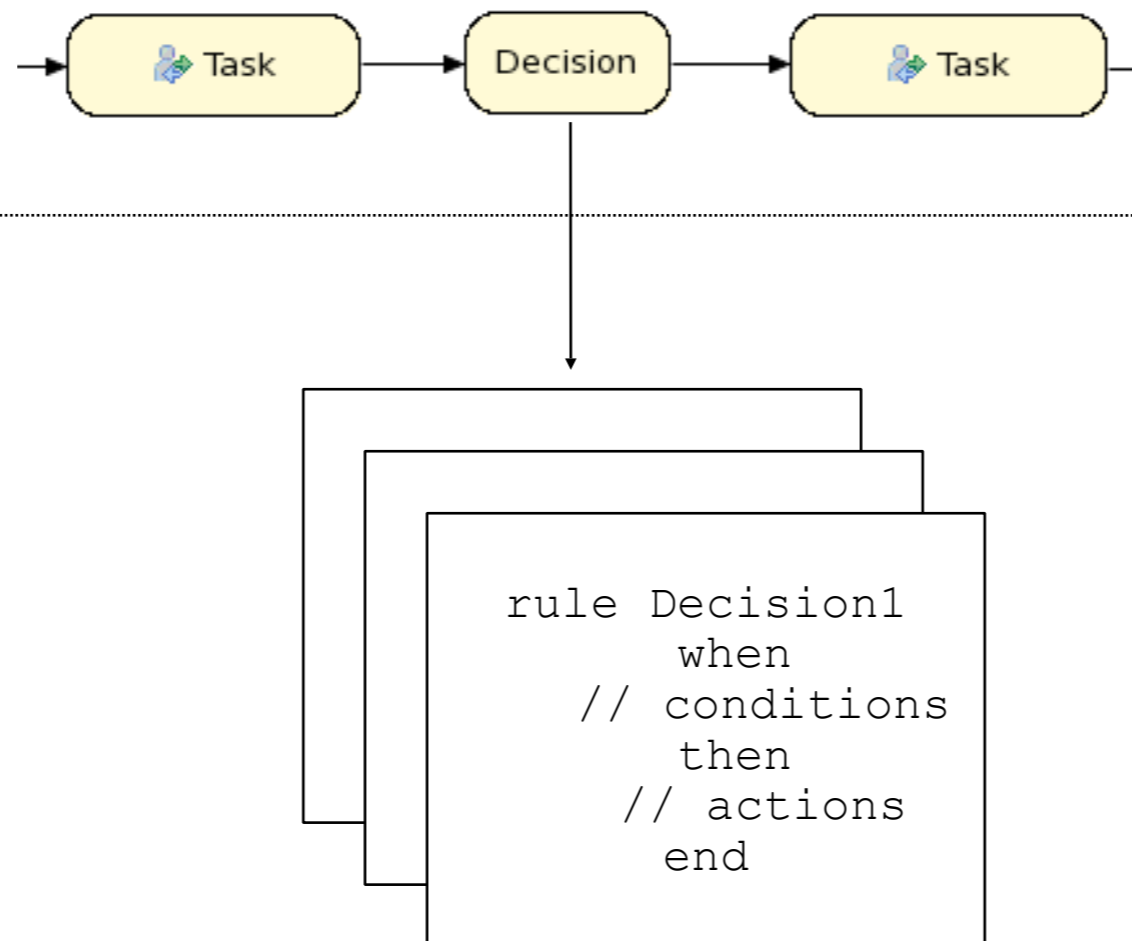
- Simplicity
- Agility
- Granularity
- Declarativeness
- Scope
- Data-centric
- Performance
- High-level
- Unification

Simplicity



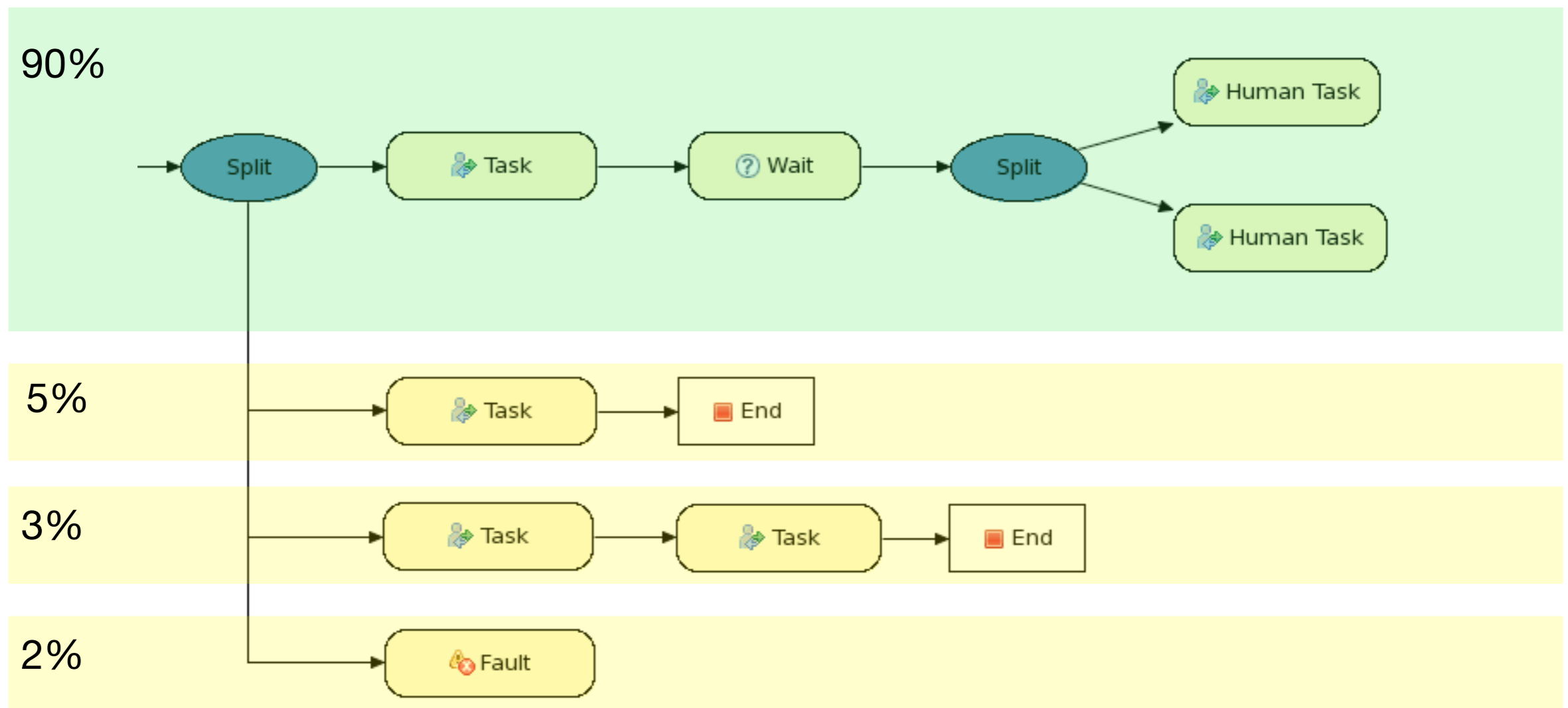
Agility

Rules and processes can have a separate life cycle



Granularity

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

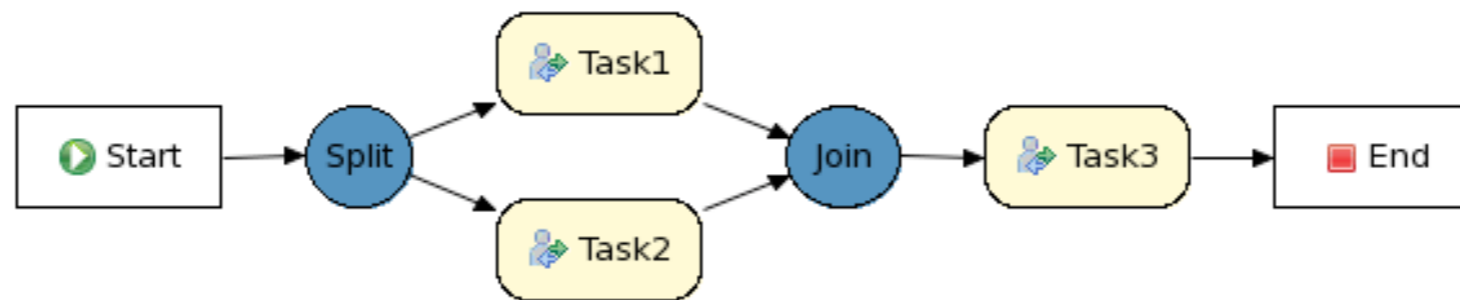


Why integrate rules and processes ?

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

How does it work?

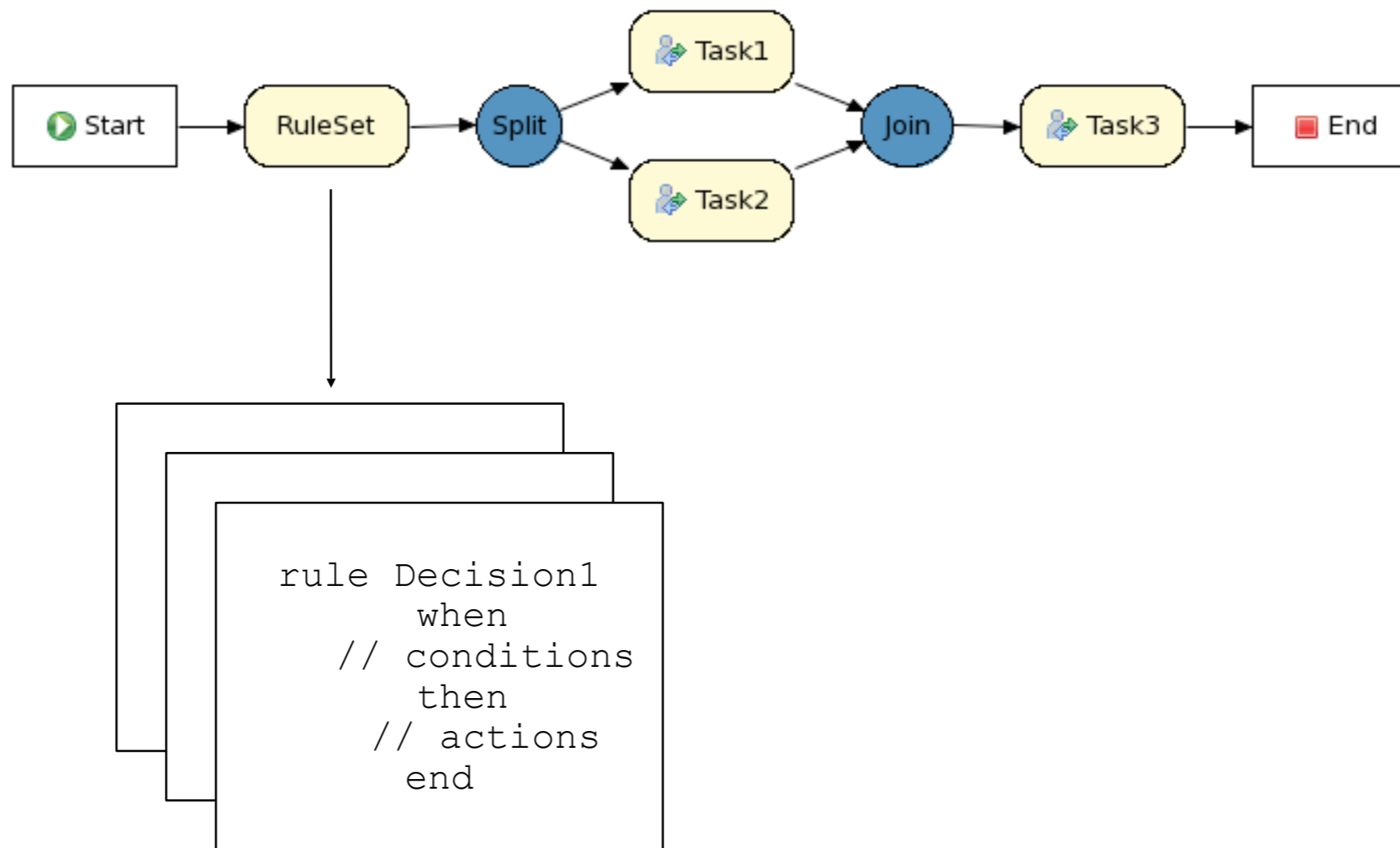
- Business decisions are hard-coded inside the processes



- Better: Extract business decision using rules

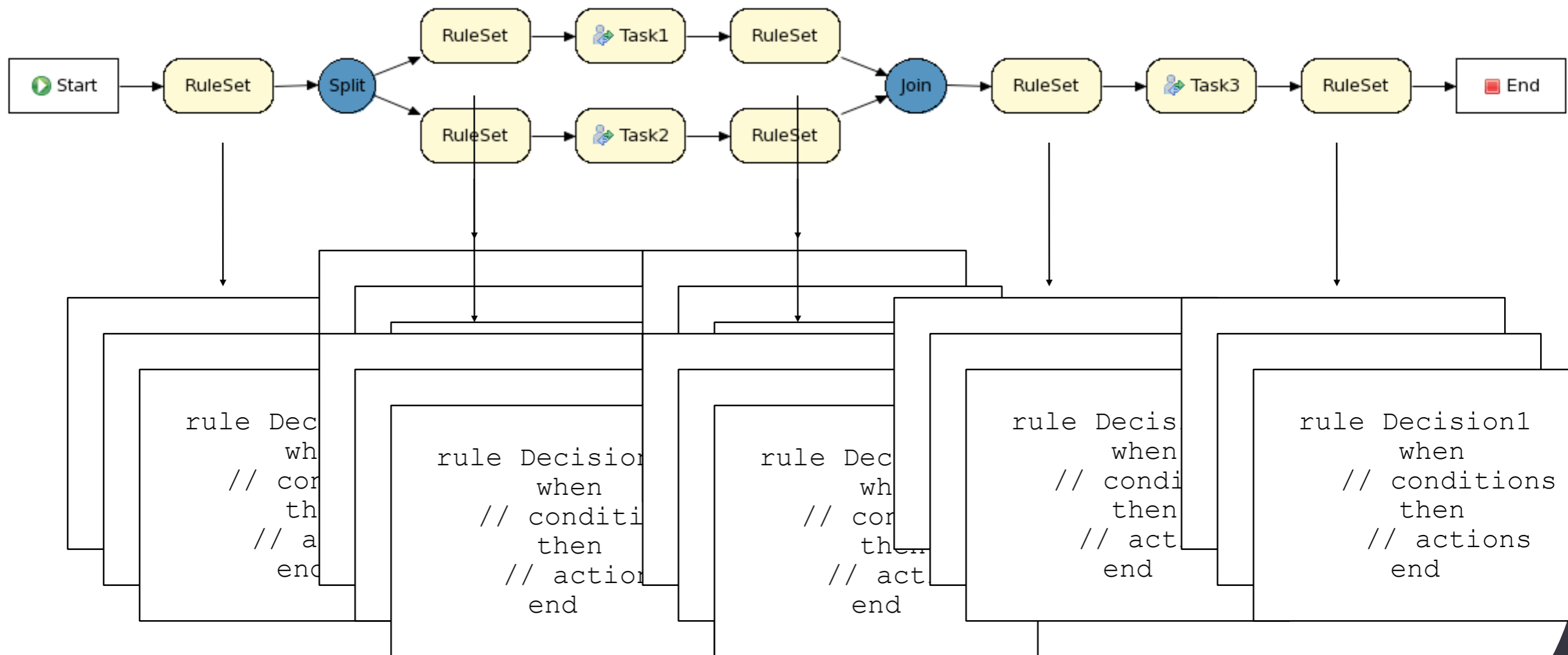
Example

- Business decisions are externalized using a decision service

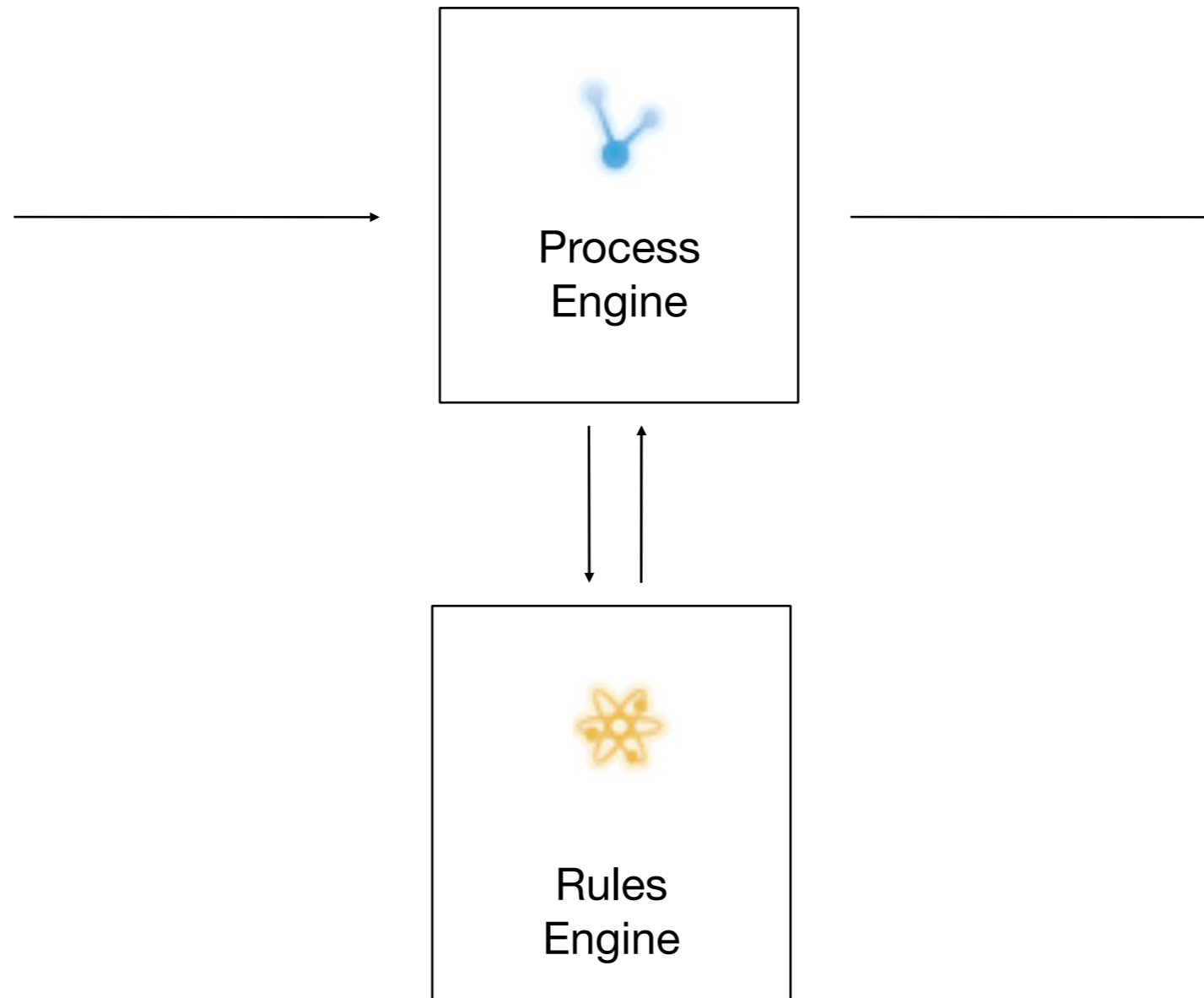


Example

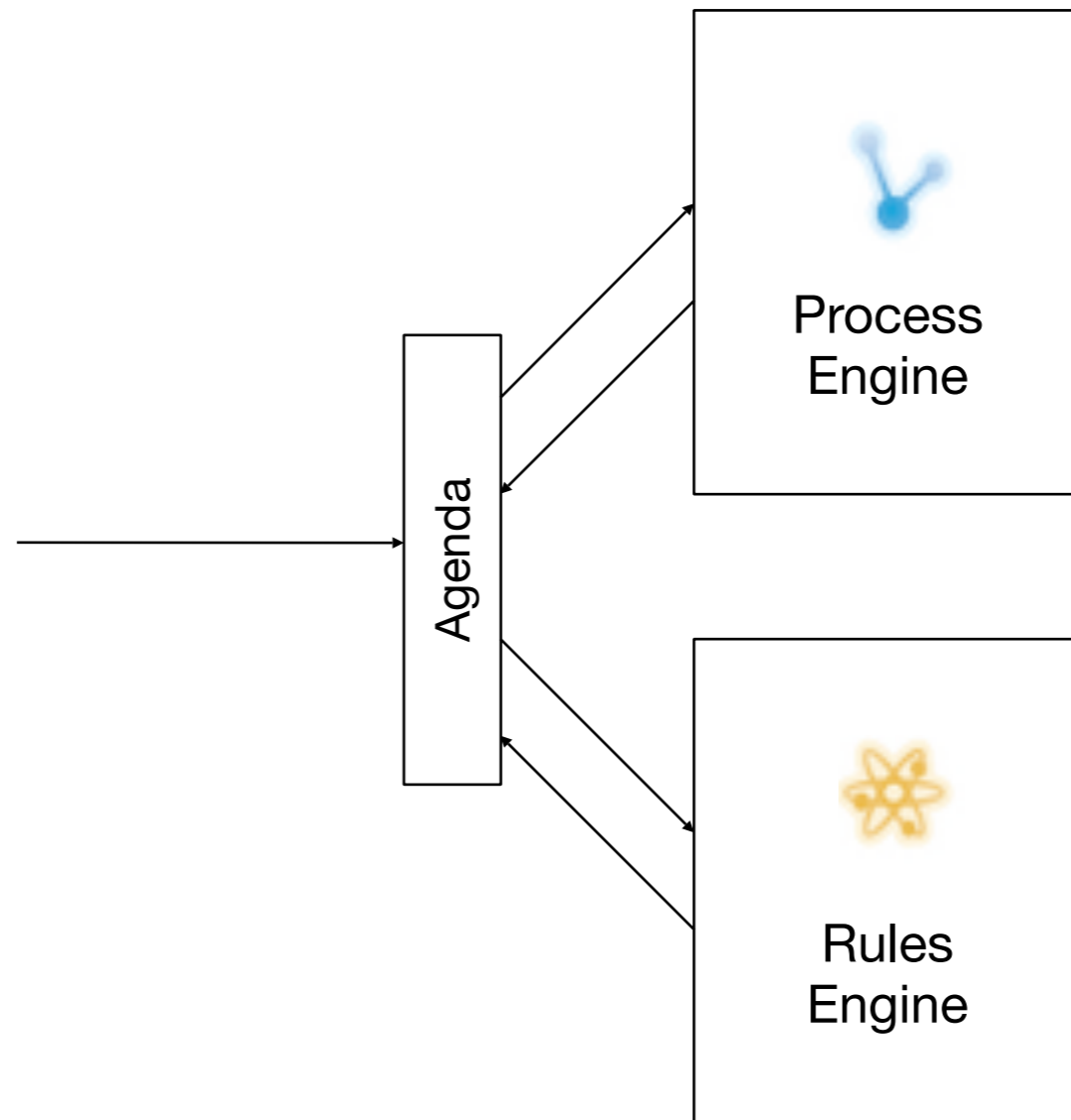
- What if there is a lot of business logic like this?



Flow of Control

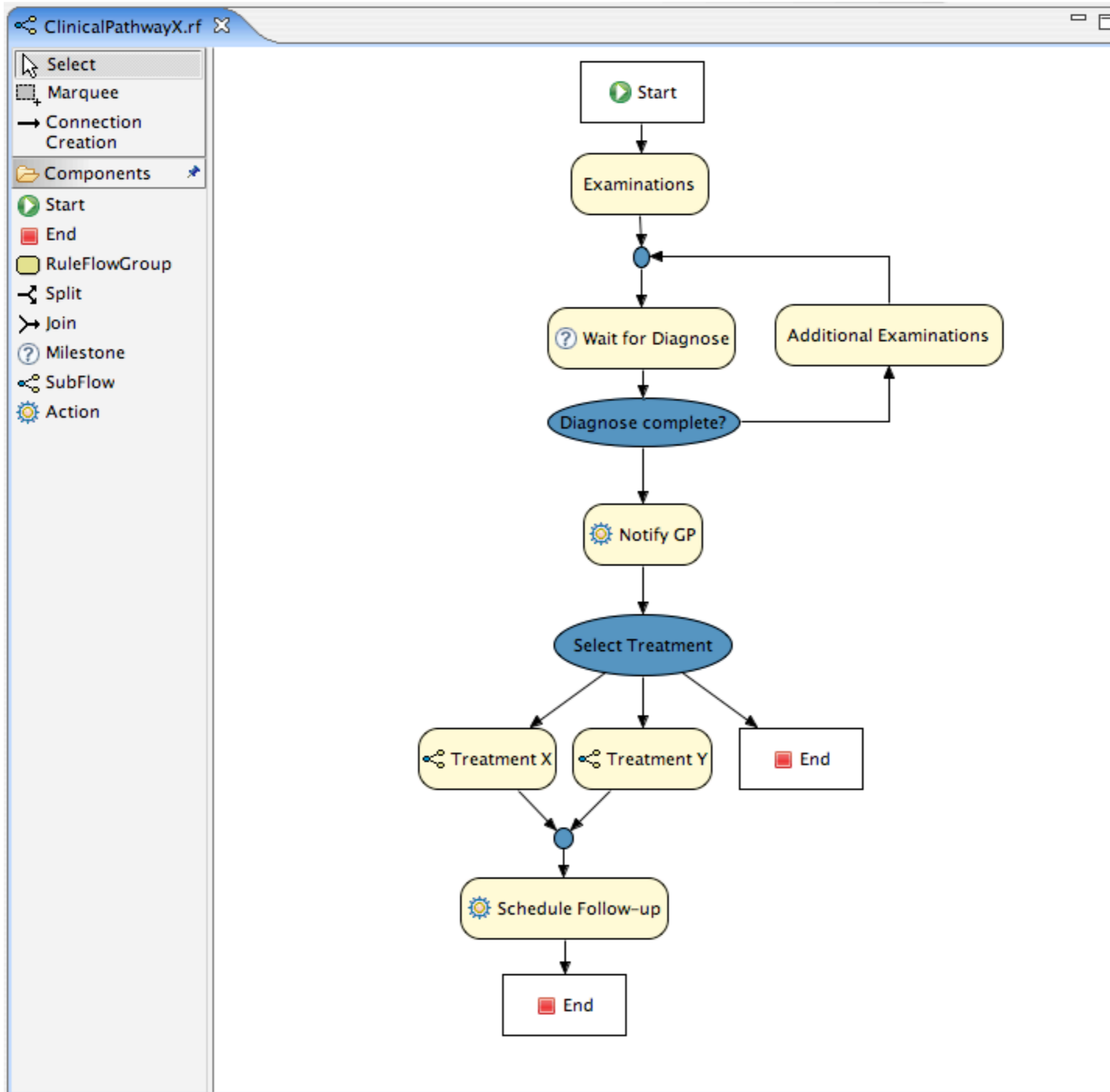


Inversion of Control



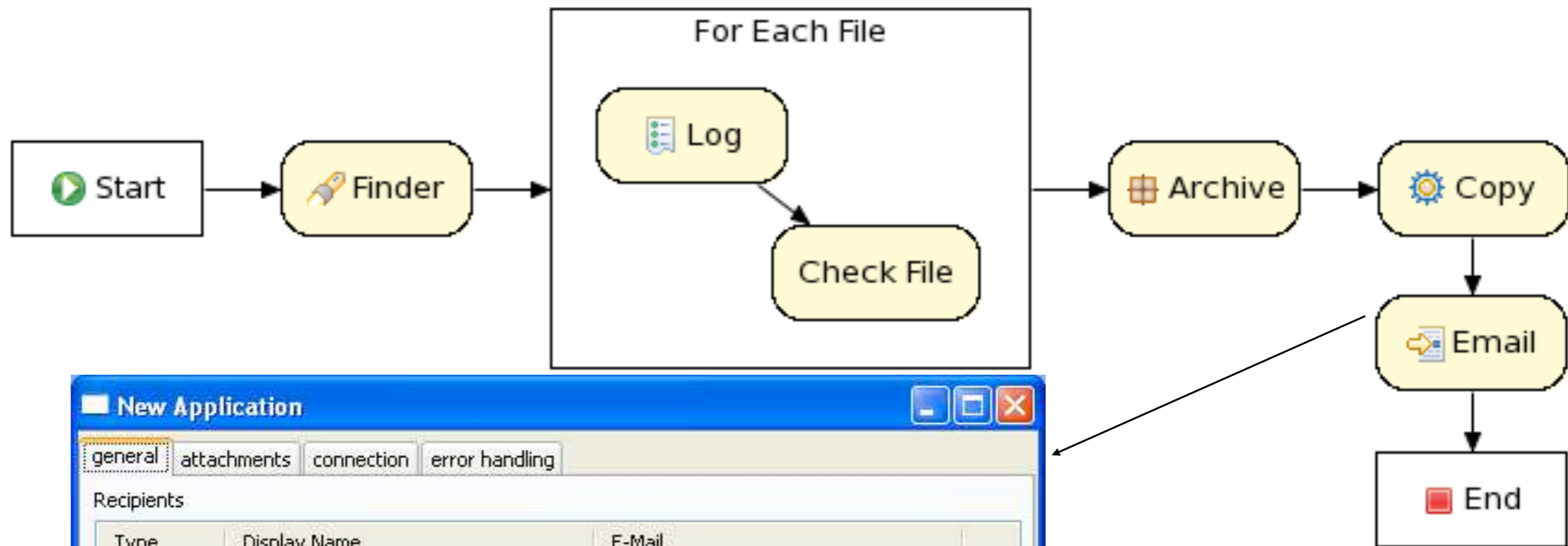
Integration Examples

- 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
- ...



Domain-specific Processes

- Select
- Marquee
- Connection Creation
- Components
- Work Items
- Email
- Exec
- Archive
- Finder
- Log



New Application

general | attachments | connection | error handling

Recipients

Type	Display Name	E-Mail
to	Tom Schindl	tom.schindl@bestsolution.at
bcc	Boris Bokowski	boris_bokowski@ca.ibm.com
cc	Tod Creasey	tod_creasey@ca.ibm.com

add delete

from:

reply to:

Subject:

type: text html

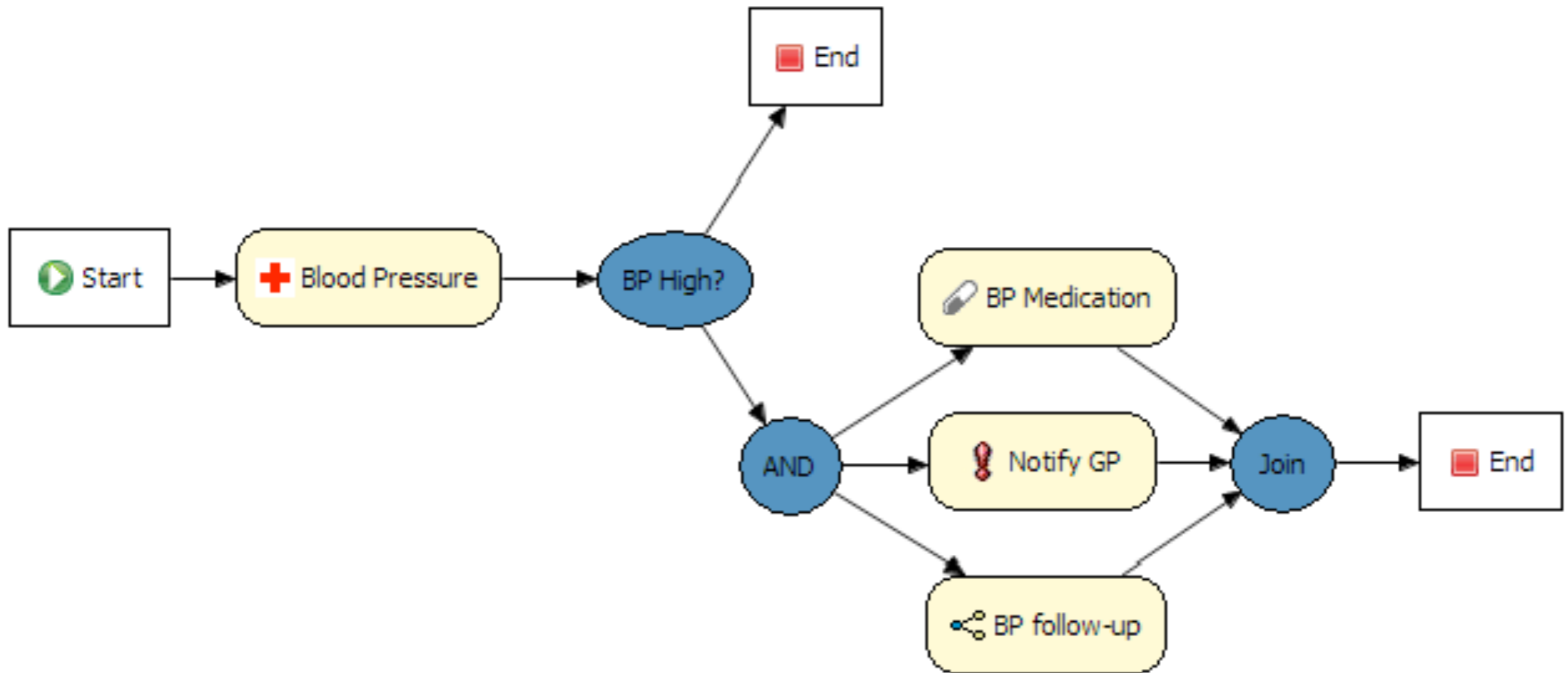
Body:

DEMO

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

```
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());
```

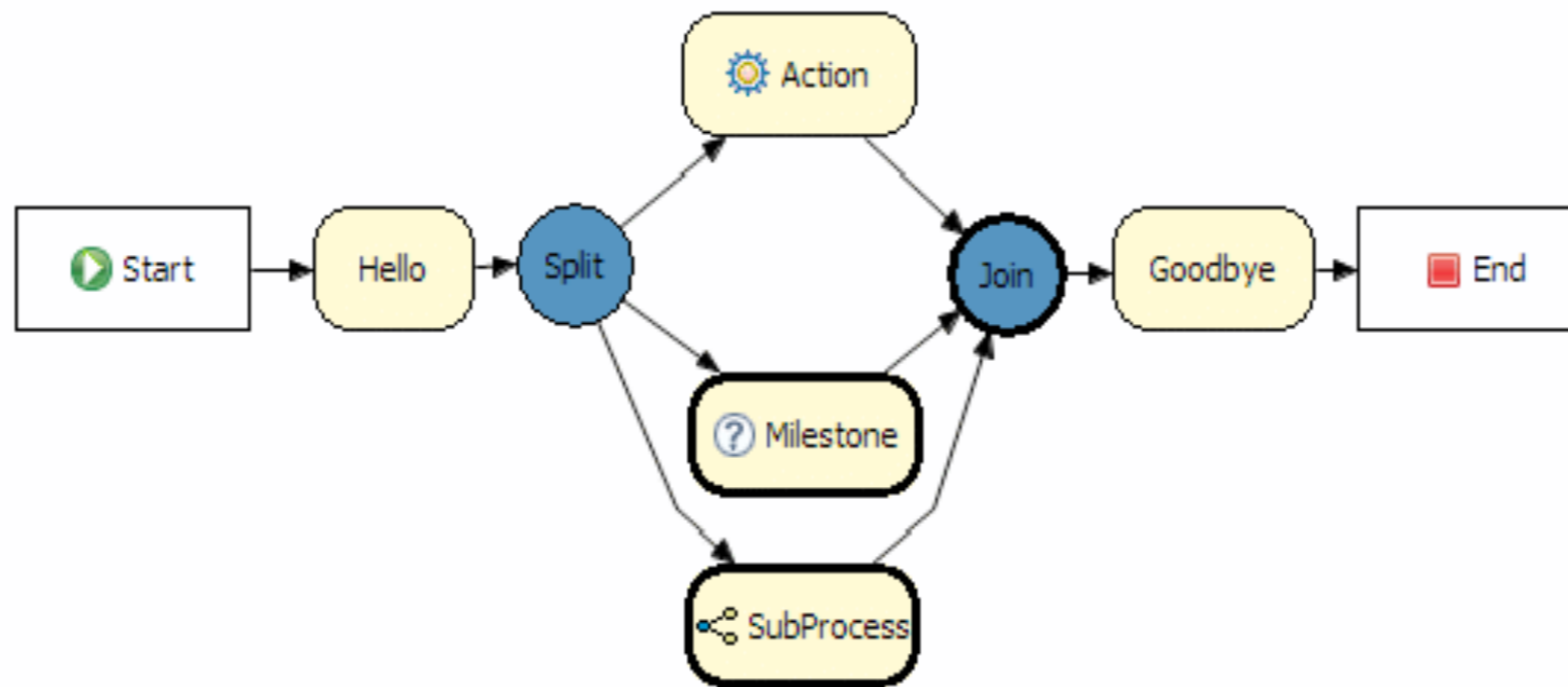
```
StatefulKnowledgeSession ksession = kbase.newStatefulKnowledgeSession();
```

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

```
ksession.fireAllRules();
```

```
ksession.startProcess("com.sample.ruleflow");
```

Integrated debug & audit



Problems Properties Audit View Error Log Console

- Object inserted (1): com.sample.RuleFlowTest\$Message@17200b4
 - Activation created: Rule Hello World m=com.sample.RuleFlowTest\$Message@17200b4(1); message=Hello World(1)
- RuleFlow started: ruleflow[com.sample.ruleflow]
 - RuleFlow node triggered: Start in process ruleflow[com.sample.ruleflow]
 - RuleFlow node triggered: Hello in process ruleflow[com.sample.ruleflow]
 - RuleFlowGroup activated: hello[size=1]
 - Activation executed: Rule Hello World m=com.sample.RuleFlowTest\$Message@17200b4(1); message=Hello World(1)
 - Object updated (1): com.sample.RuleFlowTest\$Message@17200b4
 - Activation created: Rule GoodBye m=com.sample.RuleFlowTest\$Message@17200b4(1); message=Goodbye cruel world(1)
 - RuleFlowGroup deactivated: hello[size=0]
 - RuleFlow node triggered: Goodbye in process ruleflow[com.sample.ruleflow]
 - RuleFlowGroup activated: goodbye[size=1]
 - Activation executed: Rule GoodBye m=com.sample.RuleFlowTest\$Message@17200b4(1); message=Goodbye cruel world(1)
 - RuleFlowGroup deactivated: goodbye[size=0]
 - RuleFlow node triggered: End in process ruleflow[com.sample.ruleflow]
 - RuleFlow completed: ruleflow[com.sample.ruleflow]

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

RuleFlow XML

```
<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 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

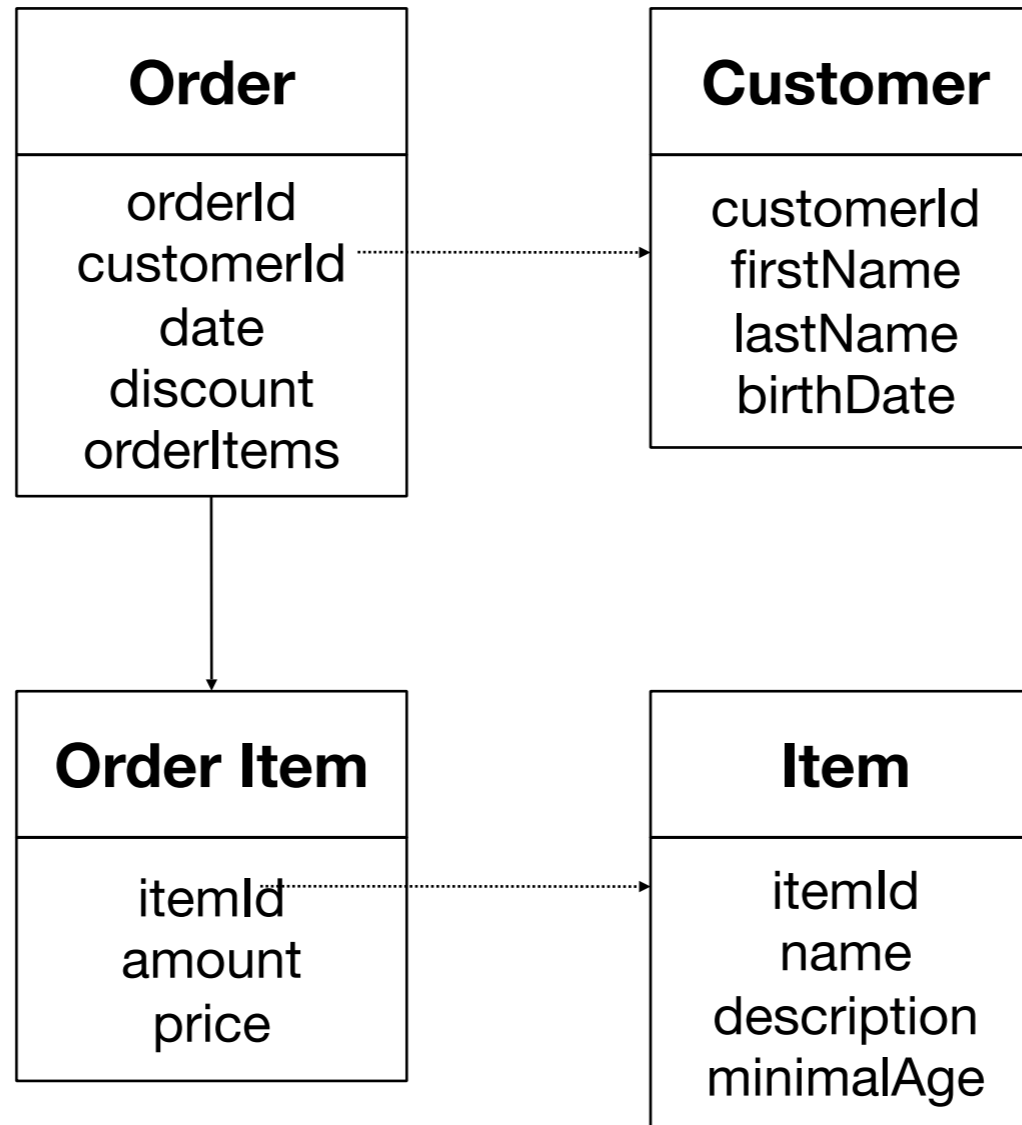
CompuSales

- 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

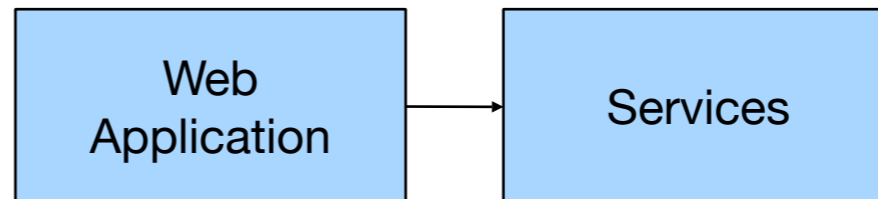
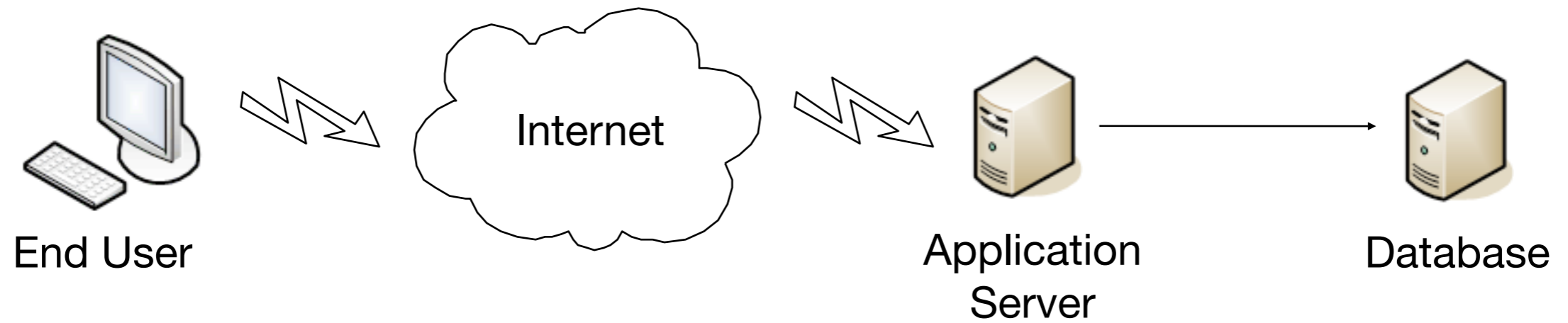
Roadmap

- 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
 - a) dynamically adapt business logic
 - b) monitor sales

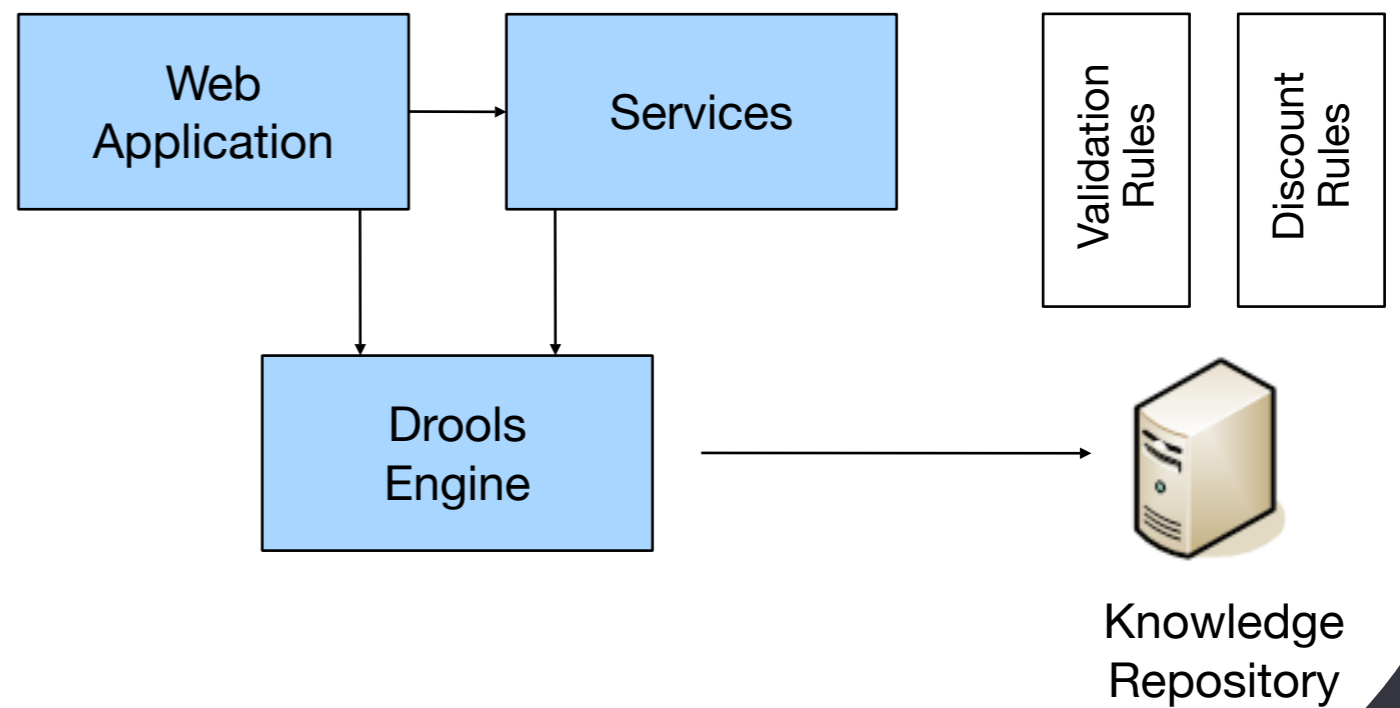
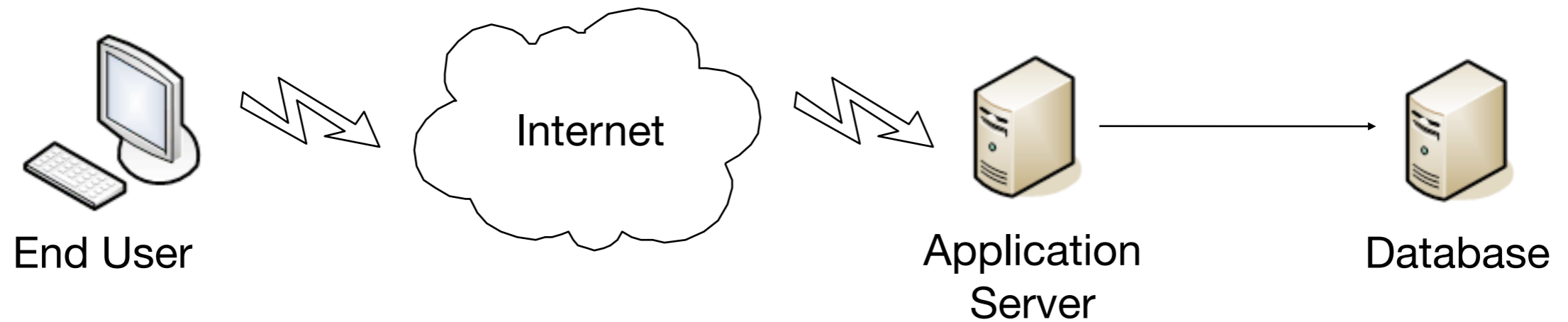
Domain Model



Overview



Step 1: 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

The screenshot displays the Drools Guvnor web interface. The top left features the Drools logo and a navigation menu. The main content area is divided into two panes. The left pane shows a tree view of the knowledge repository, with 'Business rule assets' selected. The right pane shows a list of business rules, with one rule displayed in a table.

Business rule as: [X]

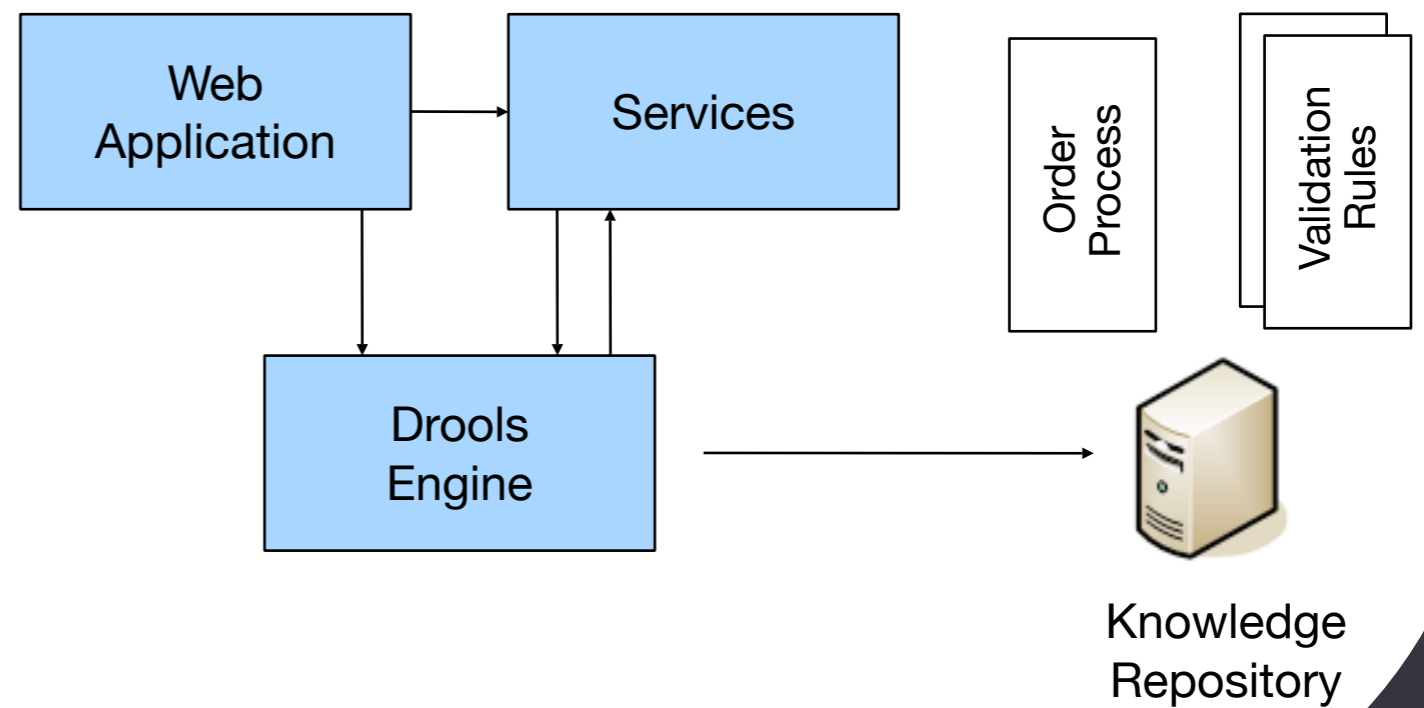
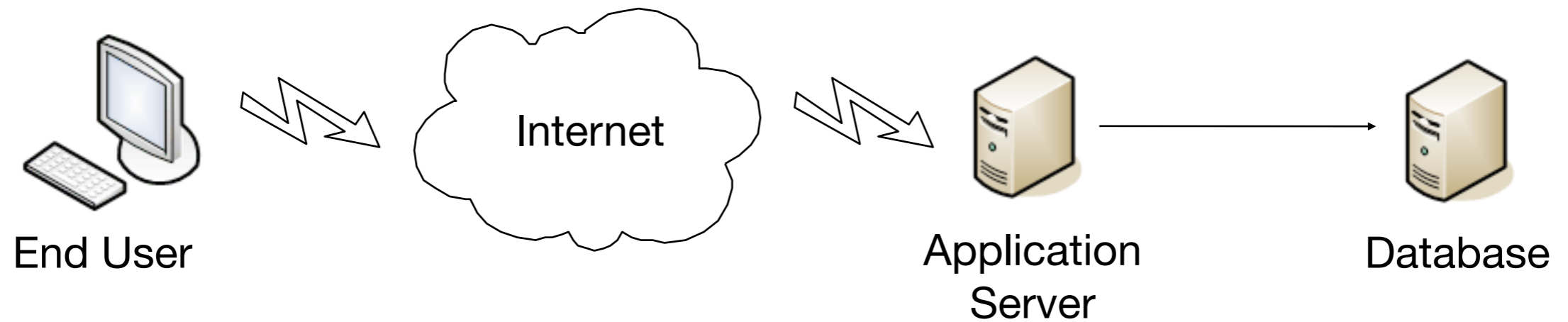
Showing #1 of 1 items. (refresh list)

Name	Last modified	Status	Categories
5% discount after 18h	Dec 6, 2008	Draft	Pricing rules

Navigation Guvnor [Navigation icons]

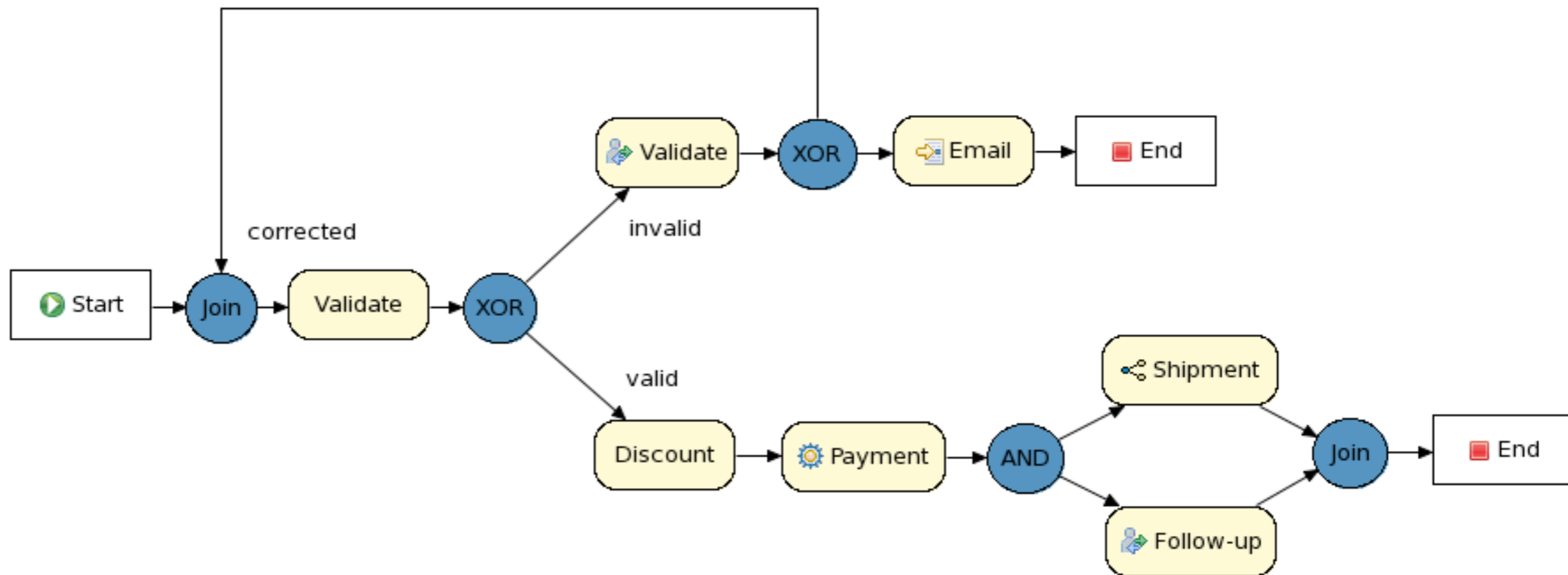
- Assets view [Expand]
- Packages [Collapse]
- Create New [Dropdown]
- Packages
 - defaultPackage
 - mortgages
 - processOrder
 - Business rule assets**
 - Technical rule assets
 - Functions
 - DSL configurations
 - Model
 - Rule Flows
 - Enumerations
 - Test Scenarios
 - XML, Properties
 - Other assets, documentati

Step 2: Processes



Step 2: Process

- Select
- Marquee
- Connection Creation
- Components
 - Start
 - End
 - RuleFlowGr...
 - Split
 - Join
 - Event Wait
 - SubFlow
 - Action
 - Timer
 - Fault
 - Event
 - Human Task
 - Composite
 - For Each
- Work Items
 - Email
 - Payment
 - Log

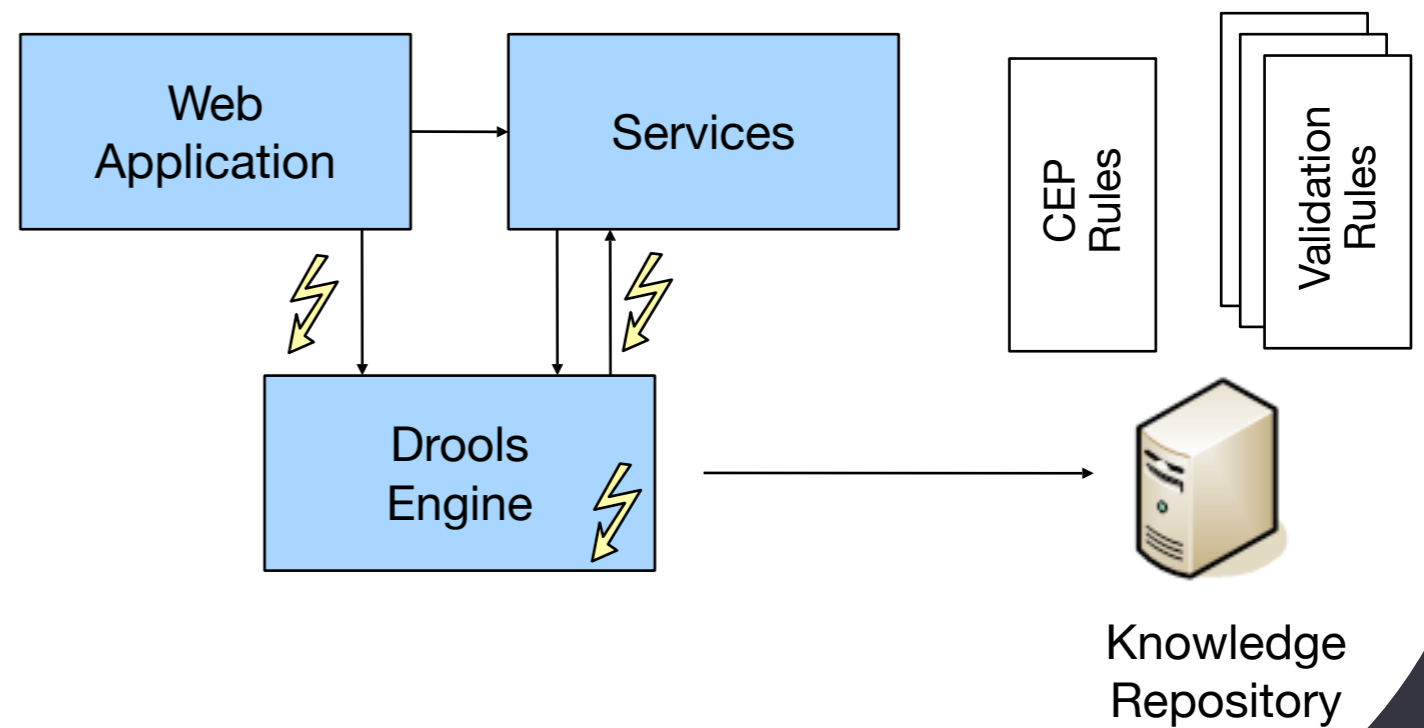
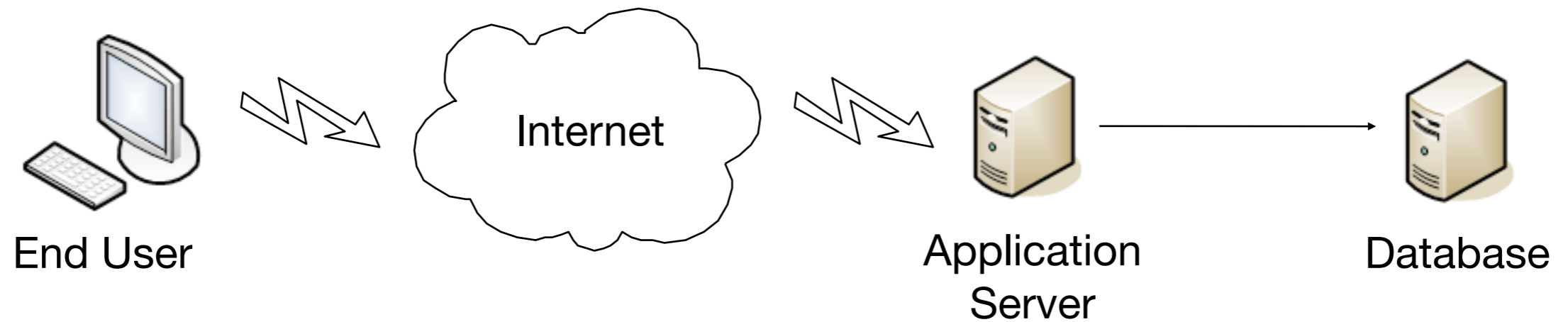


DEMO

Process and rules example



Step 3: Events



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

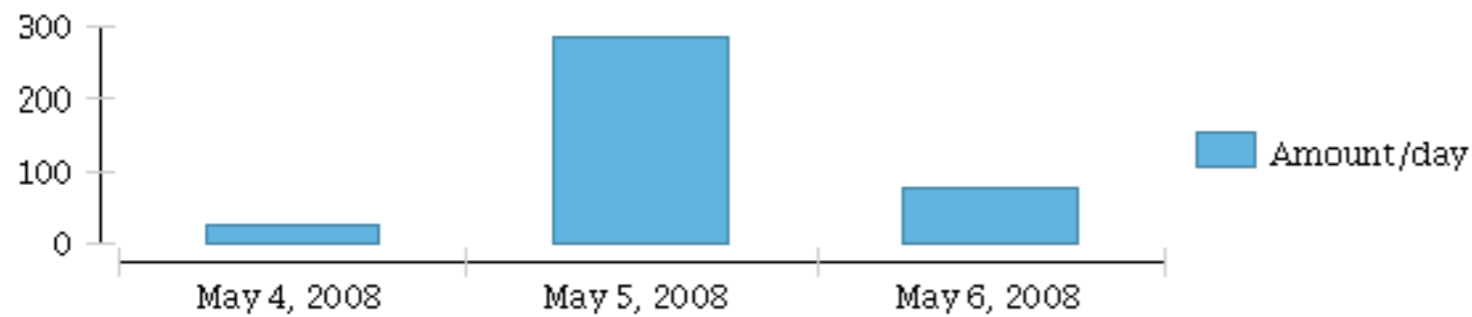
- Monitor sales
 - Use events generated by the engine (when executing rules and processes) to monitor your business
 - User-defined charts show key business indicators
 - Eclipse BIRT
 - Use CEP rules to
 - Derive higher-level events
 - Generate alerts



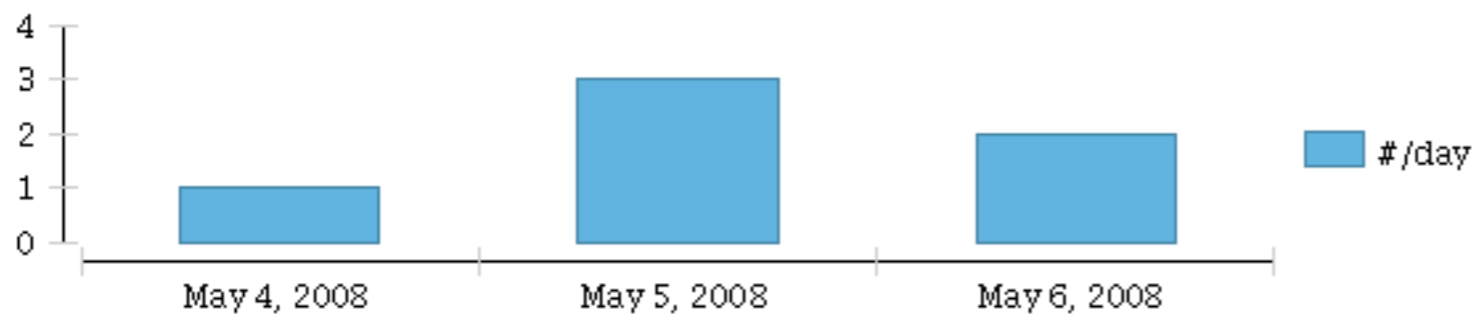
Business Activity Monitoring

Overview of current order status

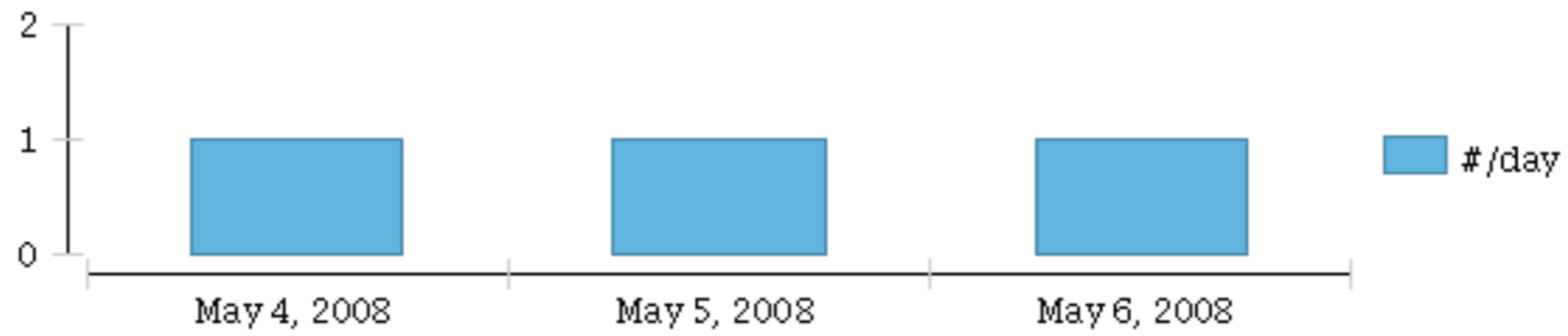
Total Amount per Day



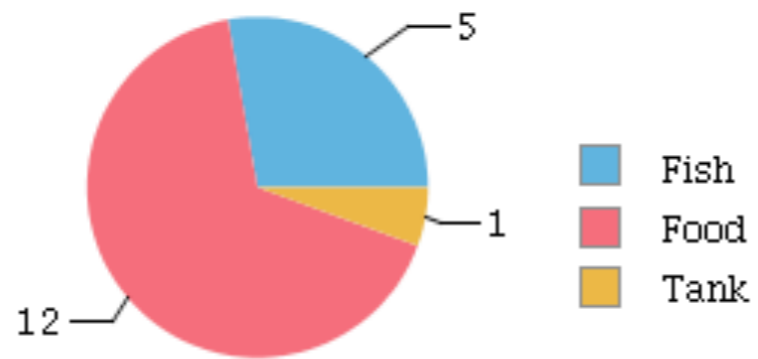
Orders per Day



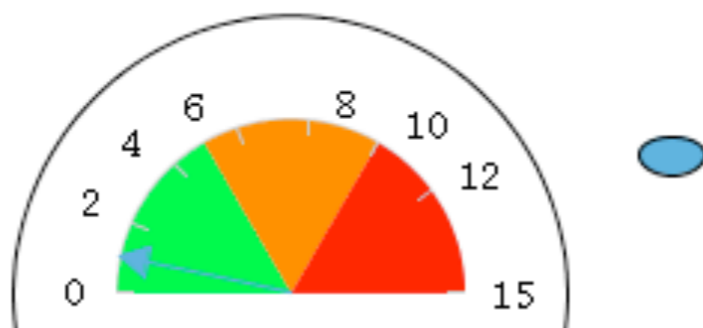
Customers per Day



Items



Outstanding Orders



Monitor sales

```
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
```

DEMO

Process, rules and events
example



Thanks for your attention!

Drools Homepage

<http://www.jboss.org/drools/>