



Guvnor 

Expert 

Fusion 

Flow 

Mark Proctor
Project Lead



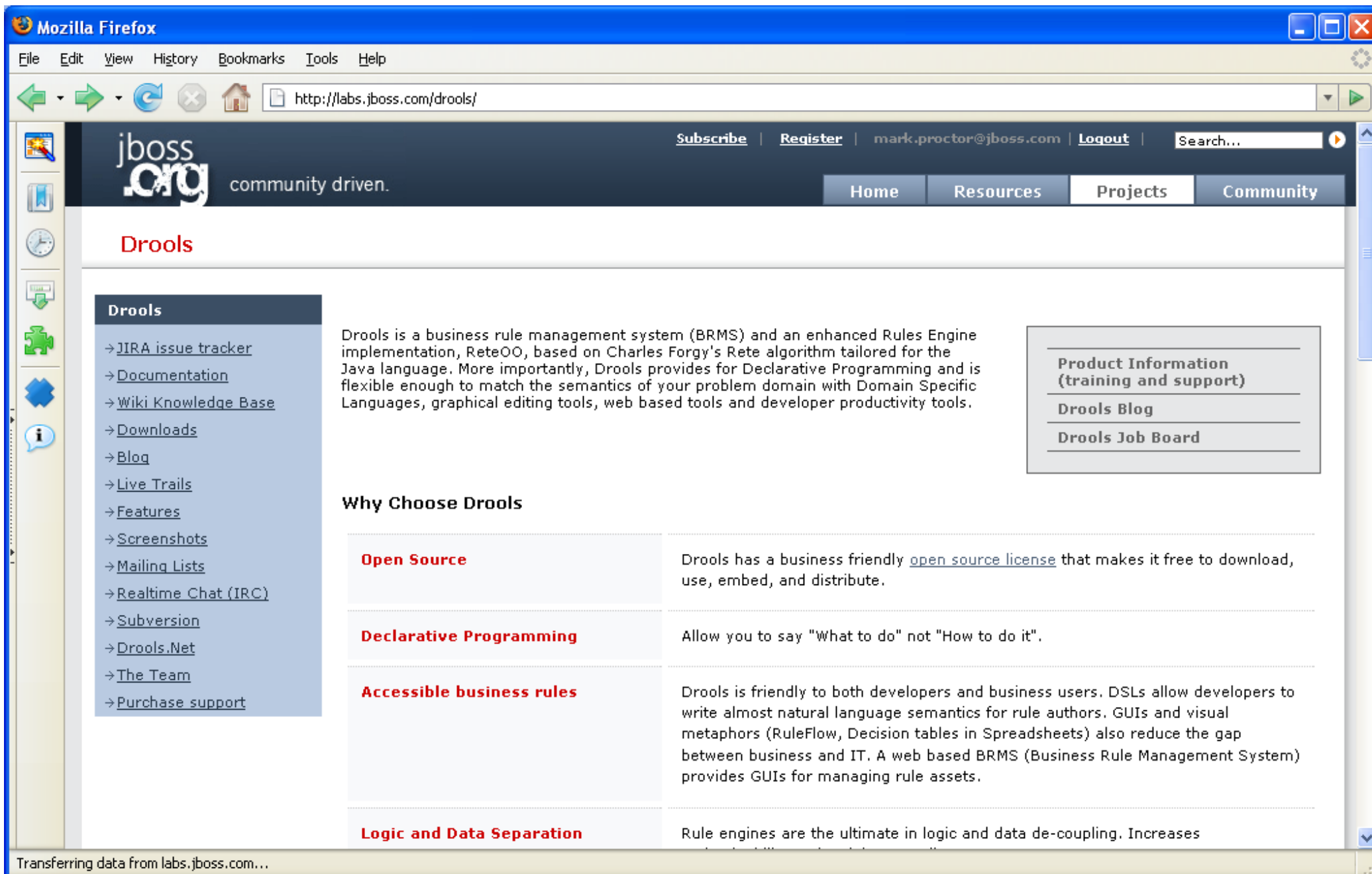
Introduction

- **The SkyNet funding bill is passed.**
- **The system goes online on August 4th, 1997.**
- **Human decisions are removed from strategic defense.**
- **SkyNet begins to learn at a geometric rate.**
- **It becomes self-aware at 2:14am Eastern time, August 29th**
- **In a panic, they try to pull the plug.**
- **And, Skynet fights back**



Drools | Agenda

- Introduction
- Expert
- Fusion
- Flow



mozilla Firefox

File Edit View History Bookmarks Tools Help

[http://labs.jboss.com/drools/](#)

jboss.org community driven.

[Subscribe](#) | [Register](#) | [mark.proctor@jboss.com](#) | [Logout](#) |

[Home](#) [Resources](#) [Projects](#) [Community](#)

Drools

Drools

- [JIRA issue tracker](#)
- [Documentation](#)
- [Wiki Knowledge Base](#)
- [Downloads](#)
- [Blog](#)
- [Live Trails](#)
- [Features](#)
- [Screenshots](#)
- [Mailing Lists](#)
- [Realtime Chat \(IRC\)](#)
- [Subversion](#)
- [Drools.Net](#)
- [The Team](#)
- [Purchase support](#)

Drools is a business rule management system (BRMS) and an enhanced Rules Engine implementation, ReteOO, based on Charles Forgy's Rete algorithm tailored for the Java language. More importantly, Drools provides for Declarative Programming and is flexible enough to match the semantics of your problem domain with Domain Specific Languages, graphical editing tools, web based tools and developer productivity tools.

Product Information (training and support)

- [Drools Blog](#)
- [Drools Job Board](#)

Why Choose Drools

Open Source	Drools has a business friendly open source license that makes it free to download, use, embed, and distribute.
Declarative Programming	Allow you to say "What to do" not "How to do it".
Accessible business rules	Drools is friendly to both developers and business users. DSLs allow developers to write almost natural language semantics for rule authors. GUIs and visual metaphors (RuleFlow, Decision tables in Spreadsheets) also reduce the gap between business and IT. A web based BRMS (Business Rule Management System) provides GUIs for managing rule assets.
Logic and Data Separation	Rule engines are the ultimate in logic and data de-coupling. Increases

Transferring data from labs.jboss.com...

Drools - Business Logic integration Platform - Mozilla Firefox

File Edit View History Bookmarks Tools Help



http://blog.athico.com/

Most Visited home Java Google KDE GNOME PostgreSQL GROKLAW Yell.com RT Radio Times Dictionary Nopaste TinyURL! jira The System R-DEVICE

Drools Boot Camp T-Shirts

Posted by Mark Proctor

T-Shirts Finally Arrived today, so everyone was very excited :) Thought I'd put up some photo's of our motley crew - Asif and Andrea (who have been here the other days) could not make it today, so they missed out on the photo. Just click any of the photos to enlarge.



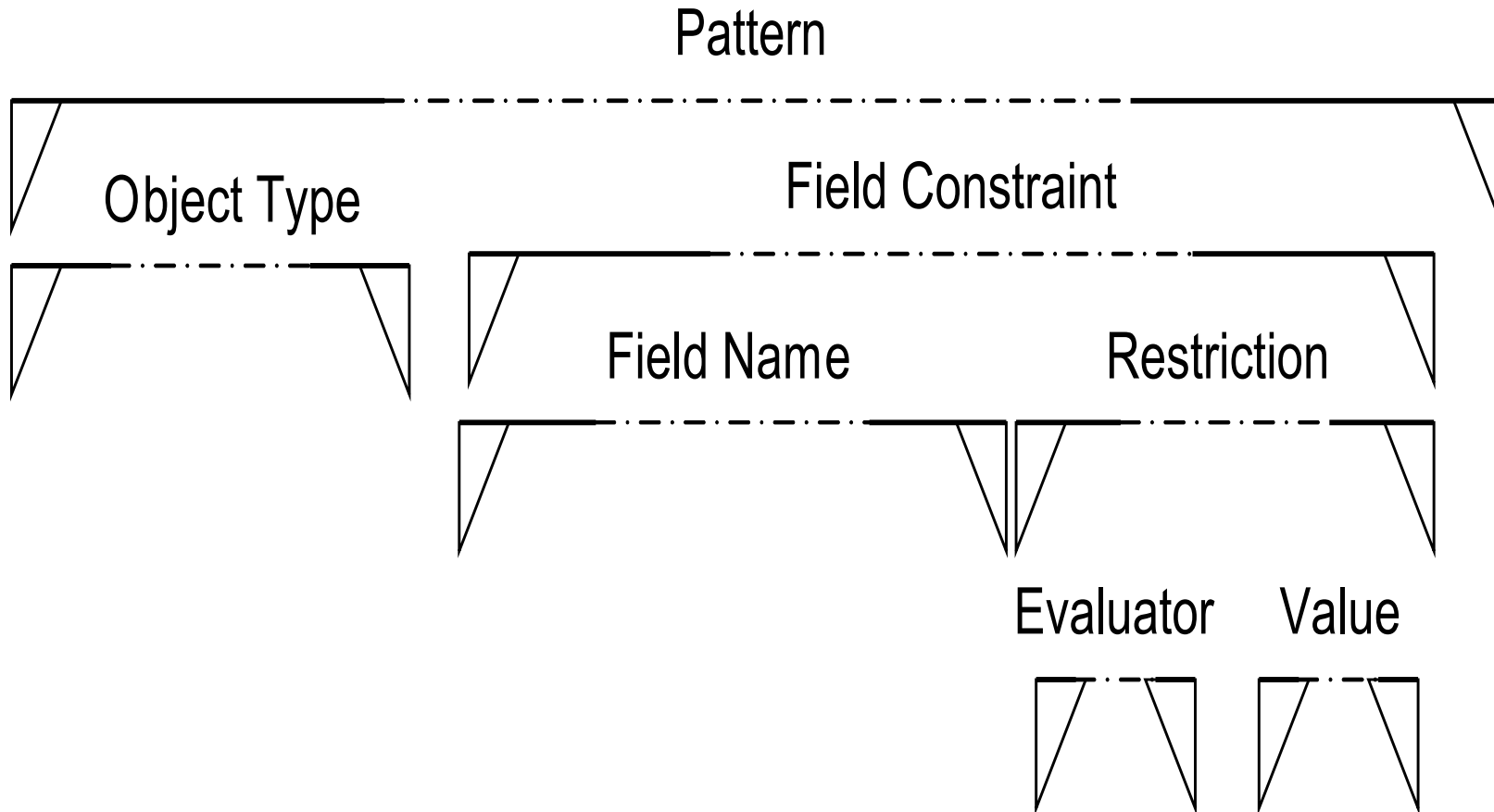
- DotNet (1)
- DRL (1)
- Drools (138)
- Drools Boot Camp (2)
- Drools Flow (5)
- drools puzzle (4)
- droos ide update-site downloads (1)
- DSL regexp antlr (1)
- DynaBeans (1)
- dynamically generated classes (1)
- Eclipse (4)
- ESP (2)
- examination (2)
- expressiveness (3)
- FactTemplate (1)
- Forward Chaining (1)
- generated classes (1)
- GIS (2)
- grammar (1)
- GSoC (3)
- GUI (6)
- Guice (1)
- Guvnor (9)
- IKVM (1)
- image processing (1)
- interview (1)
- Janino (1)
- java (2)
- JavaOne (1)
- javapolis (2)
- JBoss Rules (53)
- jBPM (2)
- JDT (1)
- Jess (2)
- JFDI (1)
- Job (4)
- JUG (2)
- KAMS (1)
- LDAP (1)
- machine learning (3)
- MicroContainer (1)
- Mind Map (1)
- MISMO (2)
- modify block (1)
- Monitoring (1)
- MVEL (6)
- MySQL (1)

Done



Drools Expert





Show(temperature == "hot")



From CE for Expressions



Drools | From CE for Expressions

Models are no longer always flat.

We need simple ways to deal with nested objects.

Drools | From CE for Expressions

```
class Person
    String name;
    List<Pet> pets;
```

```
class Pet
    String name;
    Species species
```

```
class Species
    String type;
```

Drools | From CE for Expressions

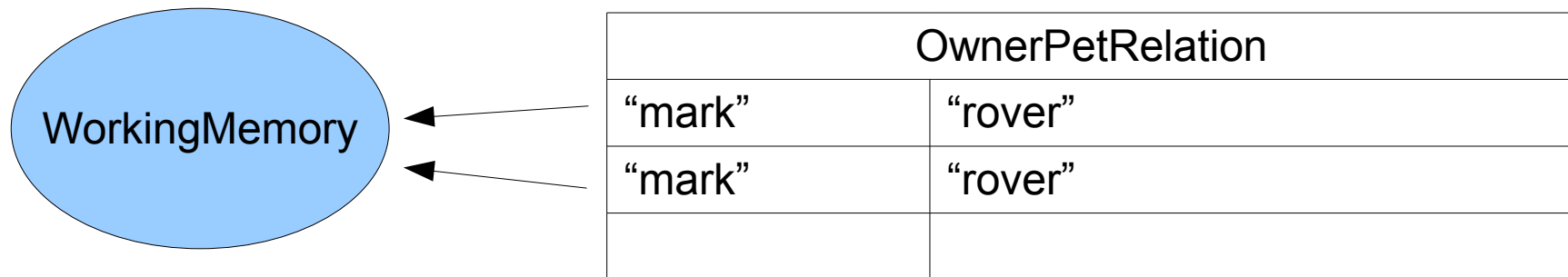
```
class OwnerPetRelation  
    String ownerName  
    String petName
```

Rule “Find all the pets for a given owner”

```
$owner : Person( name == “mark” )
```

```
OwnerPetRelationship( ownerName == $owner.name,  
                        $petName : petName )
```

```
Pet( name == $petName )
```



Drools | From CE for Expressions

But What happens if I can't create relation classes,

Or maybe I don't want to insert all nested objects.

Drools | From CE for Expressions

Patterns filter data from the working memory

Why couldn't we optionally specify the “source” of a Pattern

ObjectType() **from** expr

Drools | From CE for Expressions

Using 'from' to reason over the nested list

Rule “Find all the pets for a given owner”

when

 \$owner : Person(name == “mark”)

 Pet(name == \$petName) **from** \$owner.pets

Drools | From CE for Expressions

'from' can work on any expression, not just a nested field on a bound variable.

Rule “Find People for given zip code”

when

```
$zipCode : ZipCode()
```

```
Person( ) from $hbn.getNamedQuery(“Find People”)  
                .setParameters( [ “zipCode”, $zipCode ] )  
                .list()
```

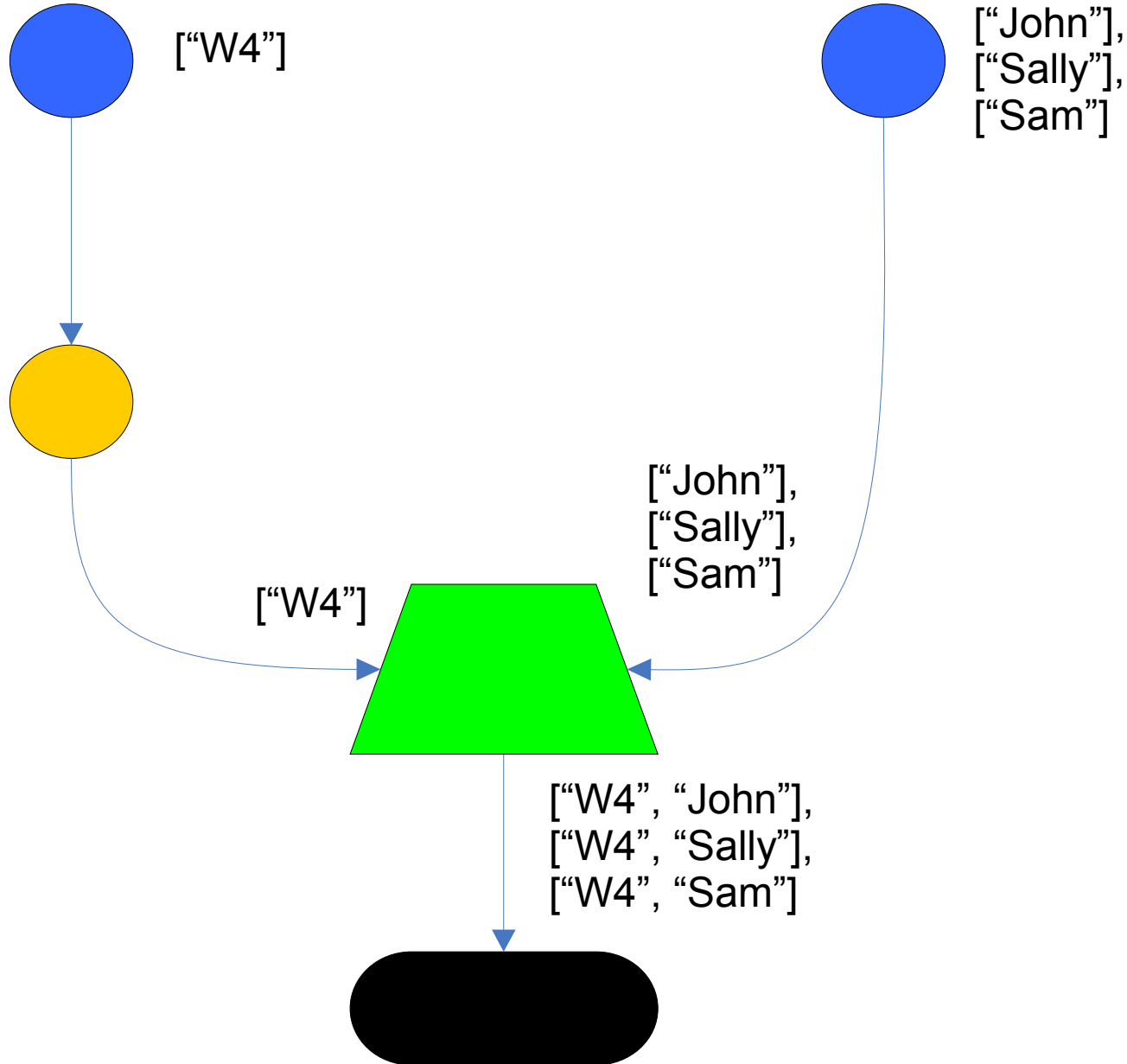


Hibernate session

Drools | From CE for Expressions

Postcode

Person



Drools | From CE for Expressions

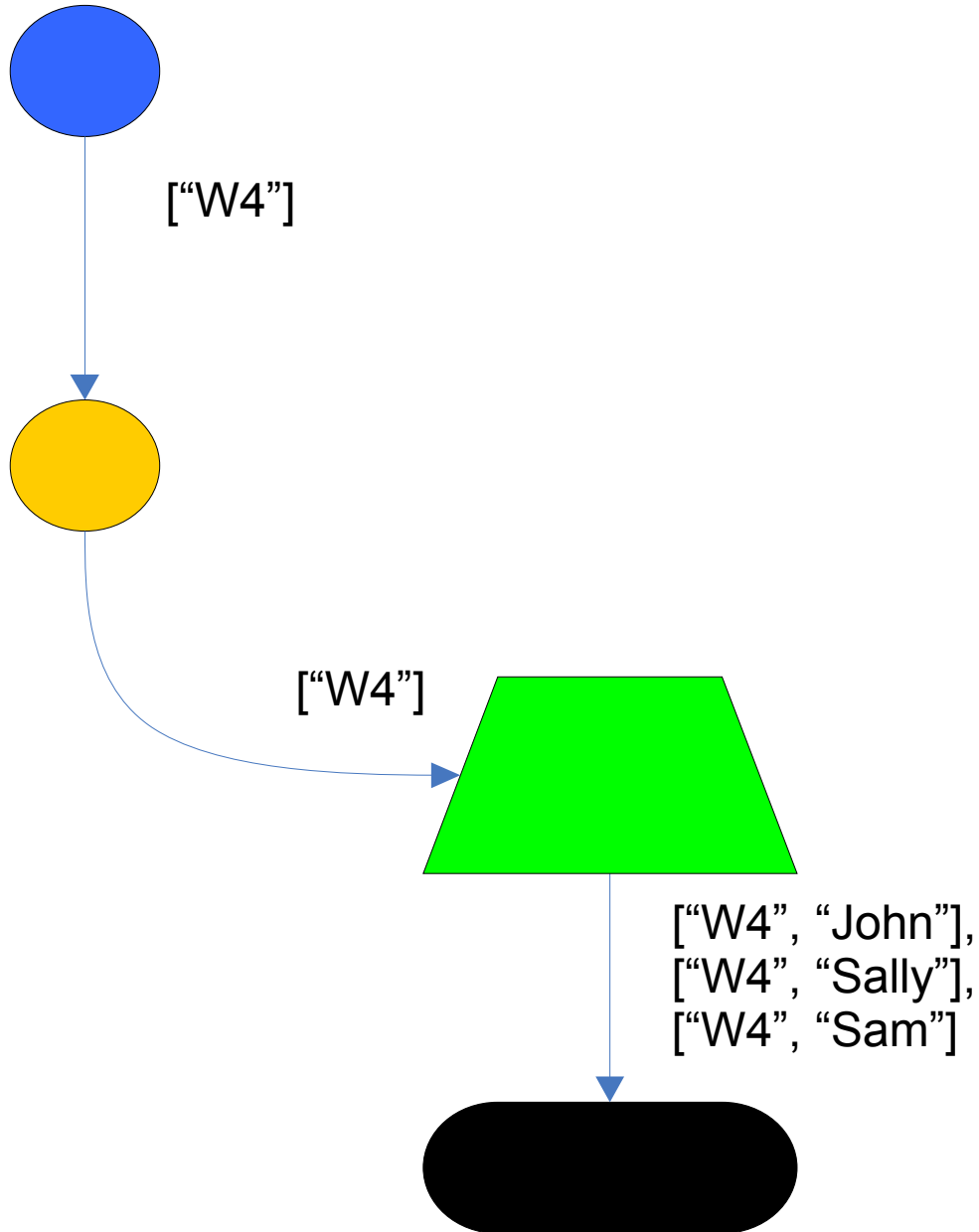
- insert “john”
 - create FactHandle
 - propagate to ObjectTypeNode
 - propagate to JoinNode.rightInput
 - wrap FactHandle with RightTuple
 - add to JoinNode.rightMemory
 - foreach (RightTuple : JoinNode.leftMemory)
 - //leftMemory is empty for this example, so no real iteration
 - create Join for “john” RightTuple with LeftTuple
 - propagate Join

Drools | From CE for Expressions

- insert “W4”
 - create FactHandle
 - propagate to ObjectTypeNode
 - propagate to LIANode
 - wrap FactHandle with LeftTuple
 - propagate to JoinNode.leftInput
 - add to JoinNode.leftMemory
 - foreach (RightTuple : JoinNode.rightMemory)
 - create Join for “W4” LeftTuple with RightTuple
 - propagate Join

Drools | From CE for Expressions

Postcode



Drools | From CE for Expressions

- insert “W4”
 - create FactHandle
 - propagate to ObjectTypeNode
 - propagate to LIANode
 - wrap FactHandle with LeftTuple
 - propagate to JoinNode.leftInput
 - foreach (RightTuple : MVEL.eval(expr,
 - LeftTuple,
context))
 - create Join for “W4” LeftTuple with RightTuple
 - propagate Join



Collect CE



- Collect all objects that match a pattern into a Collection
- evaluate the size of the collection

Drools | Collect CE

```
rule "start collect"  
then  
    insert( new Collect( "red buses",  
                        STAGE.COLLECT_START ) );  
end
```

```
rule "do collect"  
when  
    $b : Bus( colour == "red" )  
    $collect : Collect( name == "red buses",  
                       stage == STAGE.COLLECT_START )  
then  
    $collect.add( $b ); // do not notify the engine  
end
```

```
rule "end collect"  
when  
    salience -5  
    $collect : Collect( name == "red buses",  
                        stage == STAGE.COLLECT_START )  
then  
    modify( $collect ) { stage = STAGE.COLLECT_END };  
end  
rule "evaluate collect"  
when  
    $collect : Collect( size > 100,  
                        stage == STAGE.COLLECT_END )  
then  
    print "size is > 100";  
end
```

```
$var : ObjectType( constraint0...n )  
    from collect( ObjectType( constraint0...n ) )
```

```
rule "collect"  
when  
    $list : List( size > 100 )  
        from collect( Bus( color == "red" ) )  
then  
    print "size is > 100";  
end
```

Patterns and CE's can be chained with 'from'

```
rule "collect"
```

```
when
```

```
    $zipCode : ZipCode()
```

```
    $list : List( size > 100 )
```

```
        from collect( Bus( color == "red" )
```

```
            from $hbn.getNamedQuery("Find Buses")
```

```
                .setParameters( [ "zipCode",  
                                $zipCode ] )
```

```
                .list() )
```

```
then
```

```
    print "size is > 100";
```

```
end
```



Accumulate CE



Drools | Accumulate CE

- Accumulate a result for all matched patterns.
- evaluate the result



Drools | Accumulate CE

```
rule "init"  
then  
    insert( new Accumulate( "red buses",  
                            STAGE.ACC_START ) );  
end
```

```
rule "action "  
when  
    $b : Bus( colour == "red" )  
    $acc : Accumulate( name == "red buses",  
                      stage == STAGE.ACC_START )  
then  
    $acc.sum += $b.takings; // do not notify the engine  
end
```

```
rule "result"  
when  
    salience -5  
    $acc : Accumulate( name == "red buses",  
                       stage == STAGE.ACC_START )  
then  
    modify( $collect ) { stage = STAGE.ACC_END };  
end  
rule "evaluate result"  
when  
    $acc : Accumulate( size > 100, stage == STAGE.ACC_END )  
then  
    print "sum is "+ $acc.sum;  
end
```

```
$var : ObjectType( constraint0...n )  
    from accumulate( ObjectType( constraint0...n ),  
        init(...),  
        action(...),  
        result(...) )
```

Drools | Accumulate CE

```
rule "accumulate"
```

```
when
```

```
    $list : Number( intValue > 100 )
```

```
        from accumulate( Bus( color == "red", $t : takings )
```

```
            init( sum = 0 ),
```

```
            action( sum += % t ),
```

```
            result( sum ) )
```

```
then
```

```
    print "sum is "+ $acc.sum;
```

```
end
```

Drools | Accumulate CE

```
rule "accumulate"
```

```
when
```

```
    $list : Number( intValue > 100 )
```

```
        from accumulate( Bus( color == "red", $t : takings )  
                        sum( $t ) )
```

```
then
```

```
    print "sum is "+ $acc.sum;
```

```
end
```



Drools Fusion



Rule engines do not scale for CEP. They have a single point of insertion and are single threaded, CEP has concurrent streams of events.

```
session.insert( event ) ;
```

Single Point of entry

```
$c : Custerom( type == "VIP )  
BuyOrderEvent( customer == $c )
```

Patterns, evaluate facts sequentially in a single thread.

So lets allow multiple named entry points for those streams

```
EntryPoint entryPoint = session.getEntryPoint( "Home Broker Stream" );  
entryPoint.insert( event );
```

So now we can insert different streams concurrently

When not specified uses the "default" entry-point

```
$c : Custerom( type == "VIP" )  
BuyOrderEvent( customer == $c ) from entry-point "Home Broker Stream"
```

Patterns can now optional specify their entry-point.

Automatic Life-Cycle Management

All Fact life-cycles must be managed by the user, so retractions are manual.

```
declare StockTick
  @role( event )
end
```

Just use the declare statement to declare a type as an event and it will be retracted when it is no longer needed

```
declare StockTick
  @role( event )
  @timestamp( timestampAttr )
```

```
  companySymbol : String
  stockPrice : double
  timestampAttr : long
end
```

The declare statement can also specify an internal model, that external objects/xml/csv map on to. We support Smooks and JAXB

Rule engines do not have rich enough set of temporal comparison operators

```
$c : Customer( type == "VIP" )
$oe : BuyOrderEvent( customer == $c )
      from entry-point "Home Broker Stream"
BuyAckEvent( relatedEvent == $oe.id, this after[1, 10] $oe )
      from entry-point "Stock Trader Stream"
```

BackAckEvent must occur
between 1 and 10 ticks 'after'
BuyOrderEvent

The Full set of Operators are supported

- coincides
- before
- after
- meets
- metby
- overlaps
- overlappedby
- overlappedbyov
- during
- includes
- starts
- startedby
- finishes
- finishedby

Drools | Operators

```
$c : Customer( type == "VIP" )  
$oe : BuyOrderEvent( customer == $c )  
      from entry-point "Home Broker Stream"  
not BuyAckEvent( relatedEvent == $oe.id, this after[1, 10] $oe )  
      from entry-point "Stock Trader Stream"
```

Existing Drools **'not'**
Conditional Elements can
be used to detect non-
occurrence of events

Drools | Sliding time windows

Rule engines react to events happening now, there is no temporal understanding of changes over time.

StockTicker(symbol = "RHAT") over window:time(5000)

5000ms

StockTicker(symbol = "RHAT") over window:length(1000)

1000 tickers

```
$n : Number( intValue > 100 )  
    from accumulate( $s : StockTicker( symbol = "RHAT" ) over window:time( 50 ),  
                    average( $s.price ) )
```

That isn't much without the ability to deal with aggregations, rules engines suck.

Rule Engines do not deal with aggregations

```
$n : Number( intValue > 100 )
```

```
from accumulate( $s : StockTicker( symbol = "RHAT" ) over window:time( 5000 ),  
                average( $s.price ) )
```

The pattern 'Number'
reasons 'from' the
accumulate result

Aggregate ticker price
for RHAT over last 5
seconds

Over 5 seconds

```
$n : accumulate( $s : StockTicker( symbol = "RHAT" ) over window:time( 5000 ),  
                average( $s.price ) > 100 )
```

We can use some sugar to reduce
verbosity

```
$n : accumulate( $s : StockTicker( symbol = "RHAT" ) over window:time( 50 ),  
[ min( $s ) > 100, max( $s ) < 250 ] )
```

Binds to a list of
the two values

We can even work with
multiple value
aggregations



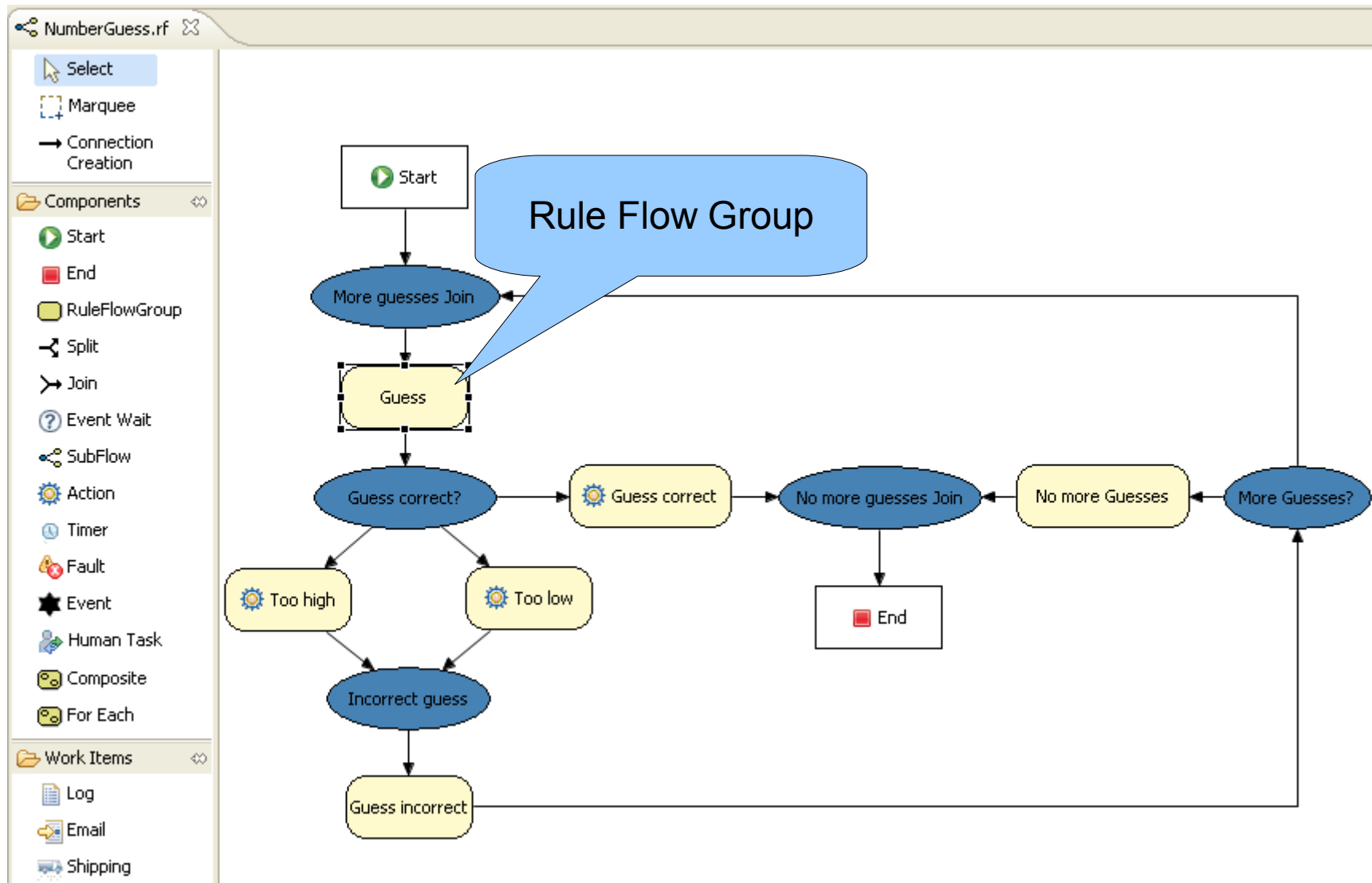
Drools Flow



Workflow can control my rules?

```
rule "Get user Guess"  
  ruleflow-group "Guess"  
  no-loop  
  when  
    $r : RandomNumber()  
    rules : GameRules( allowed : allowedGuesses )  
    game : Game( guessCount < allowed )  
    not ( Guess() )  
  then  
    System.out.println( "You have " + ( rules.allowedGuesses - game.guessCount ) )  
    br = new BufferedReader( new InputStreamReader( System.in ) )  
    modify ( game ) { guessCount += 1 }  
    i = br.readLine()  
    insert( new Guess( i ) )  
  end
```

Drools RuleFlowGroup



Drools RuleFlowGroup

NumberGuess.rf

Select
Marquee
Connection Creation

Components

- Start
- End
- RuleFlowGroup
- Split
- Join
- Event Wait
- SubFlow
- Action
- Timer
- Fault
- Event
- Human Task
- Composite
- For Each

Work Items

- Log
- Email
- Shipping

```
graph TD; Start([Start]) --> Join([More guesses Join]); Join --> Guess[Guess]; Guess --> Correct{Guess correct?}; Correct --> ActionCorrect[Guess correct]; Correct --> Incorrect{Incorrect guess}; Incorrect --> TooHigh[Too high]; Incorrect --> TooLow[Too low]; TooHigh --> Incorrect2{Incorrect guess}; TooLow --> Incorrect2; Incorrect2 --> ActionIncorrect[Guess incorrect]; ActionIncorrect --> Join;
```

NumberGuess.drl

```
package org.drools.examples

dialect "xml"

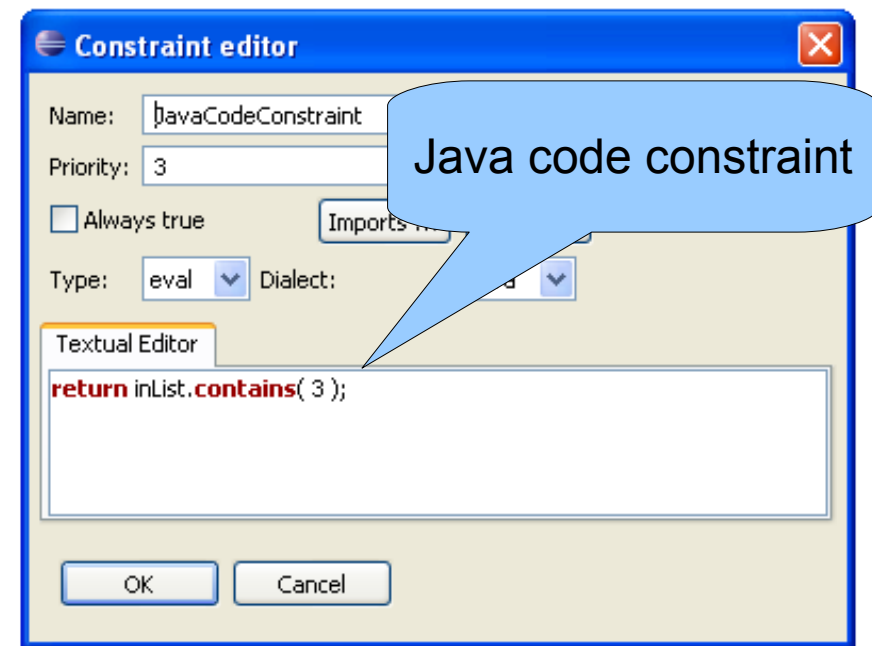
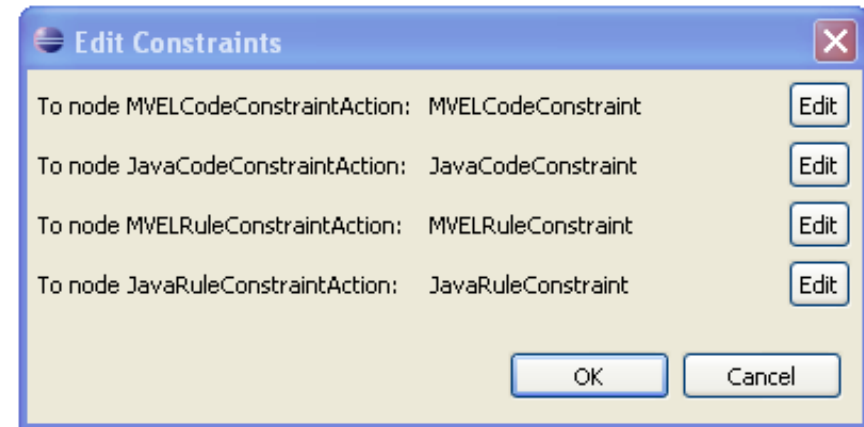
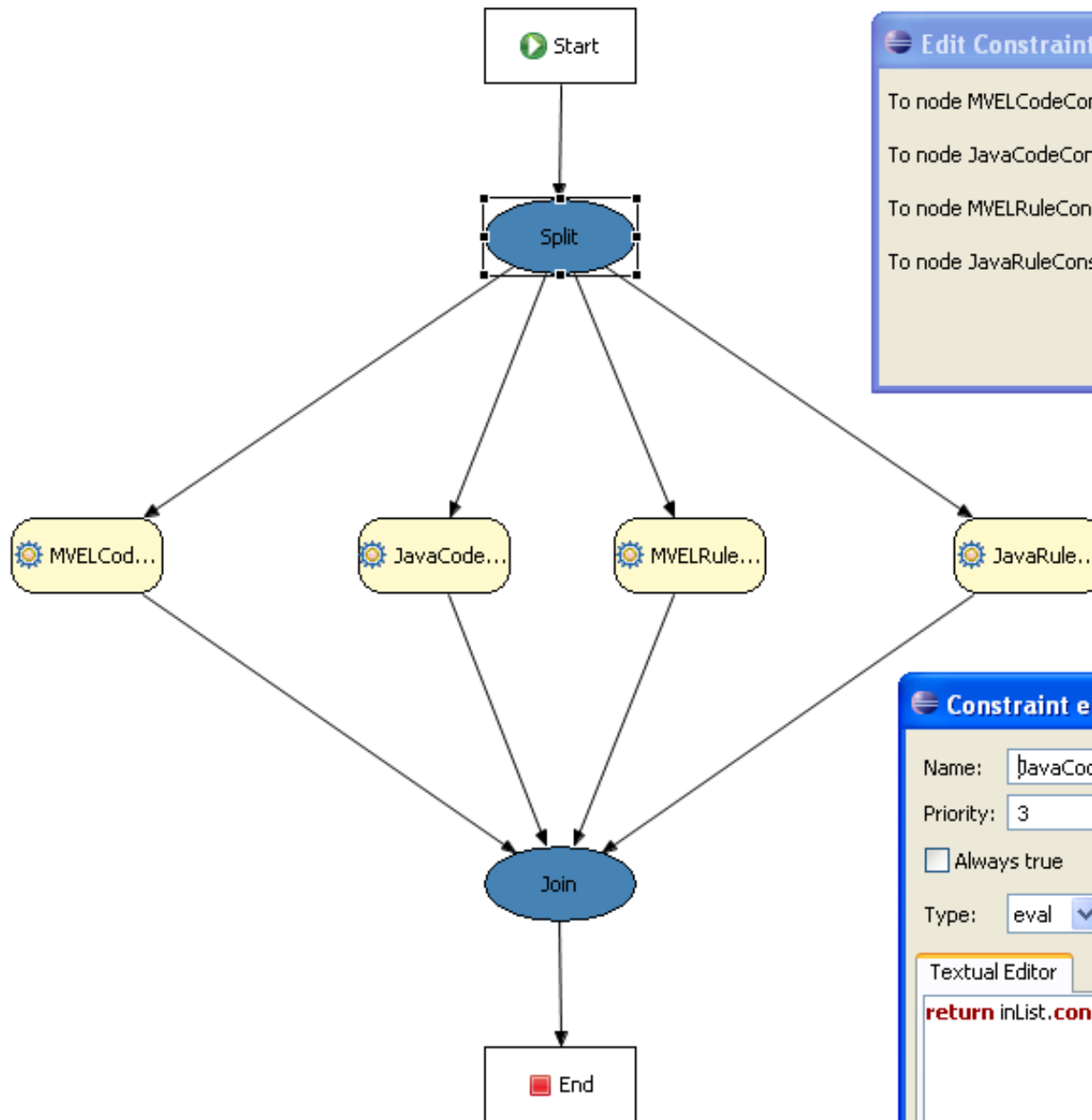
import org.drools.examples.NumberGuessExample.RandomNumber
import org.drools.examples.NumberGuessExample.Guess
import org.drools.examples.NumberGuessExample.Game
import org.drools.examples.NumberGuessExample.GameRules

import java.io.InputStreamReader;
import java.io.BufferedReader;

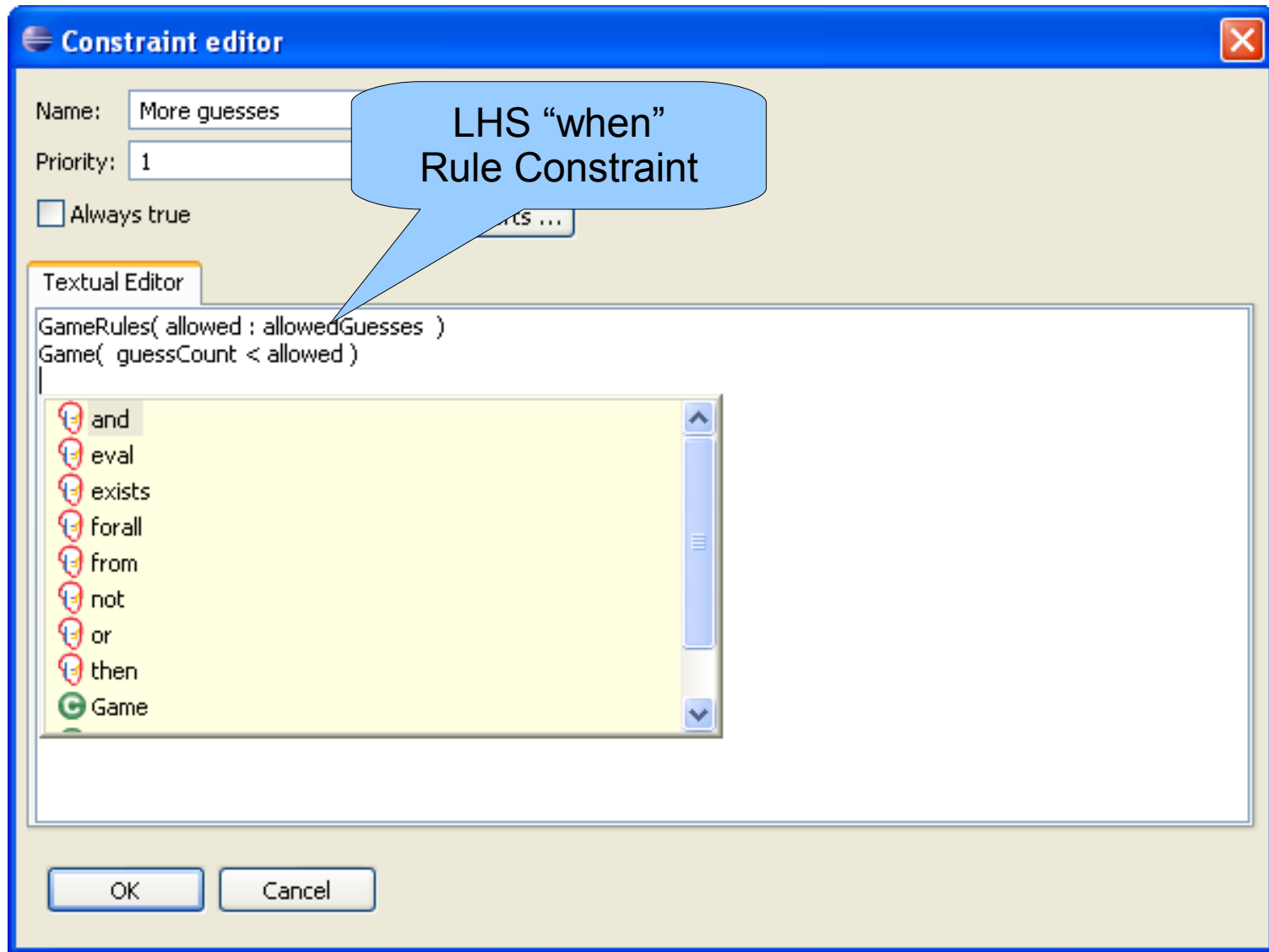
rule "Get user Guess"
  ruleflow-group "Guess"
  no-loop
  when
    $r : RandomNumber()
    rules : GameRules( allowed : allowedGuesses )
    game : Game( guessCount < allowed )
    not ( Guess() )
  then
    System.out.println( "You have " + ( rules.allowedGuesses
    br = new BufferedReader( new InputStreamReader( System
    modify ( game ) { guessCount += 1 )
    i = br.readLine();
    insert( new Guess( i ) );
  end

rule "Record the biggest Guess"
  ruleflow-group "Guess"
  no-loop
  when
    game : Game( biggestGuess : biggest )
    Guess( $value : value > biggestGuess )
  then
    modify ( game ) { biggest = $value };
  end
```

Drools Constraints

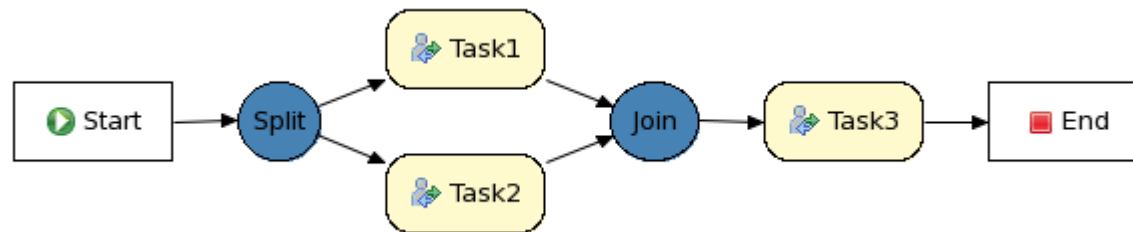


Rules can control my workflow?



Example (1)

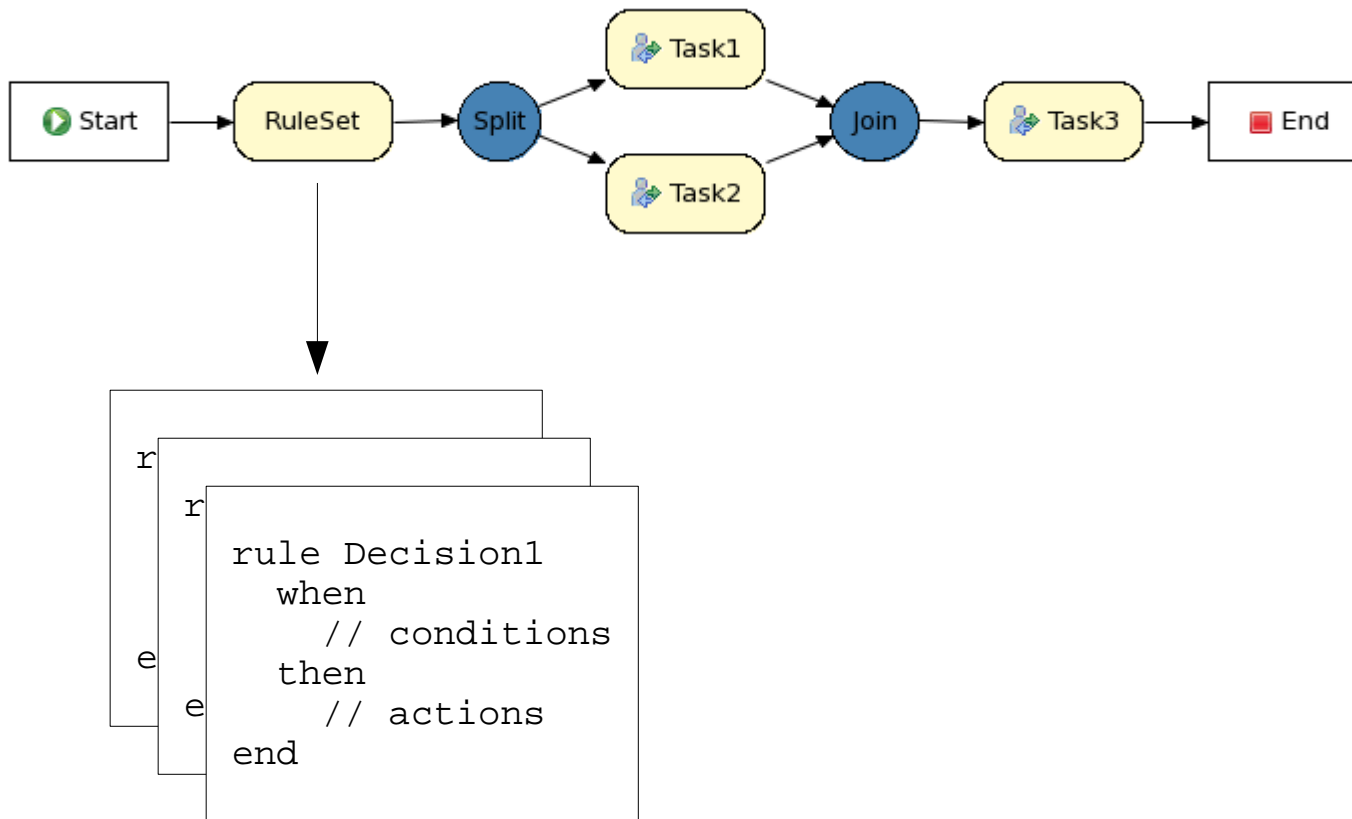
- Business decisions are hard-coded inside the processes



- Better: Extract business decision using rules

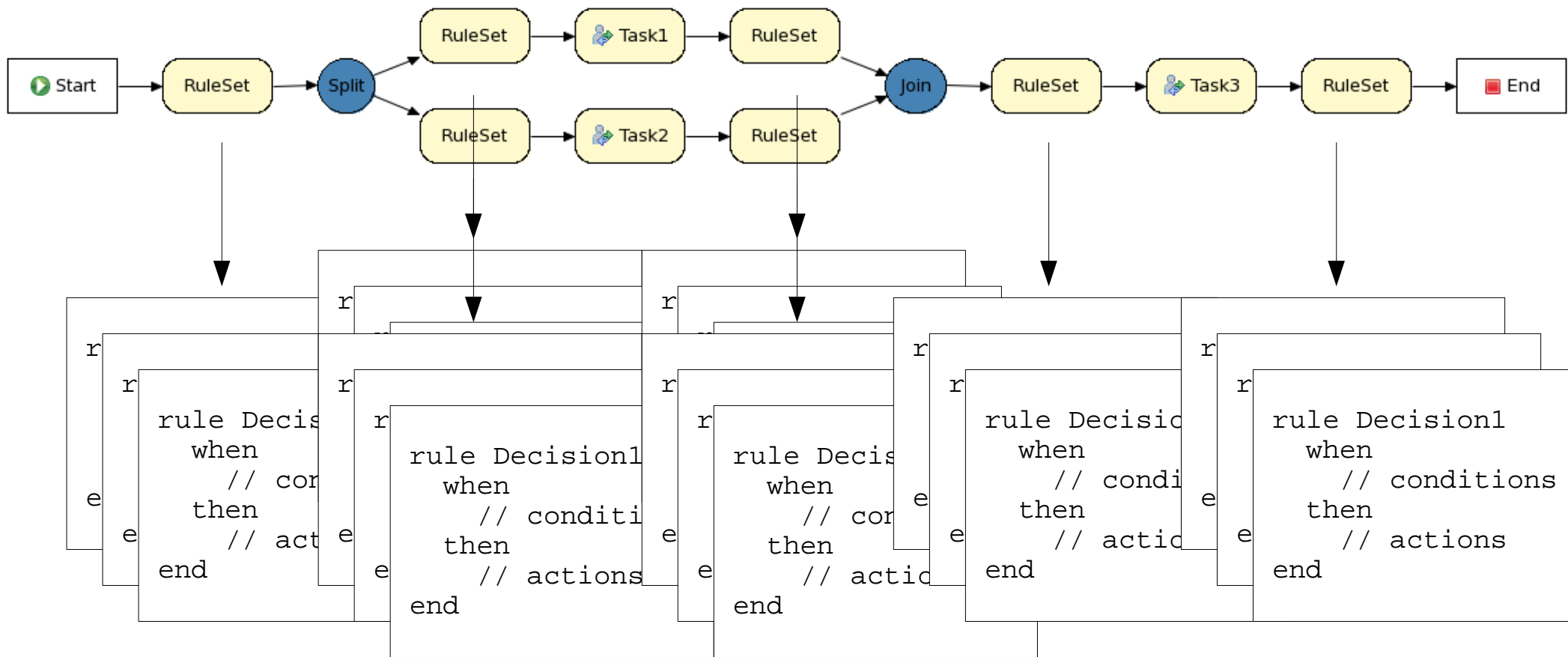
Example (2)

- Business decisions are externalized using a decision service

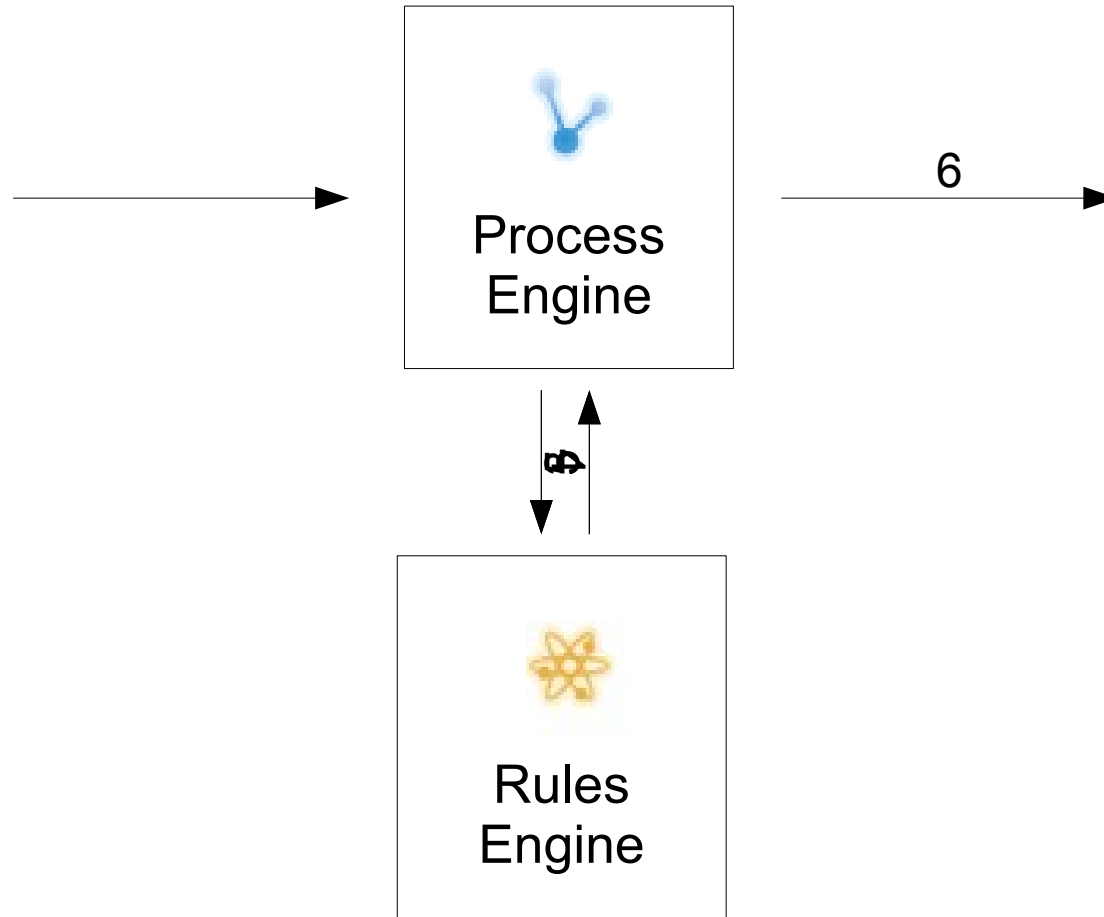


Example (3)

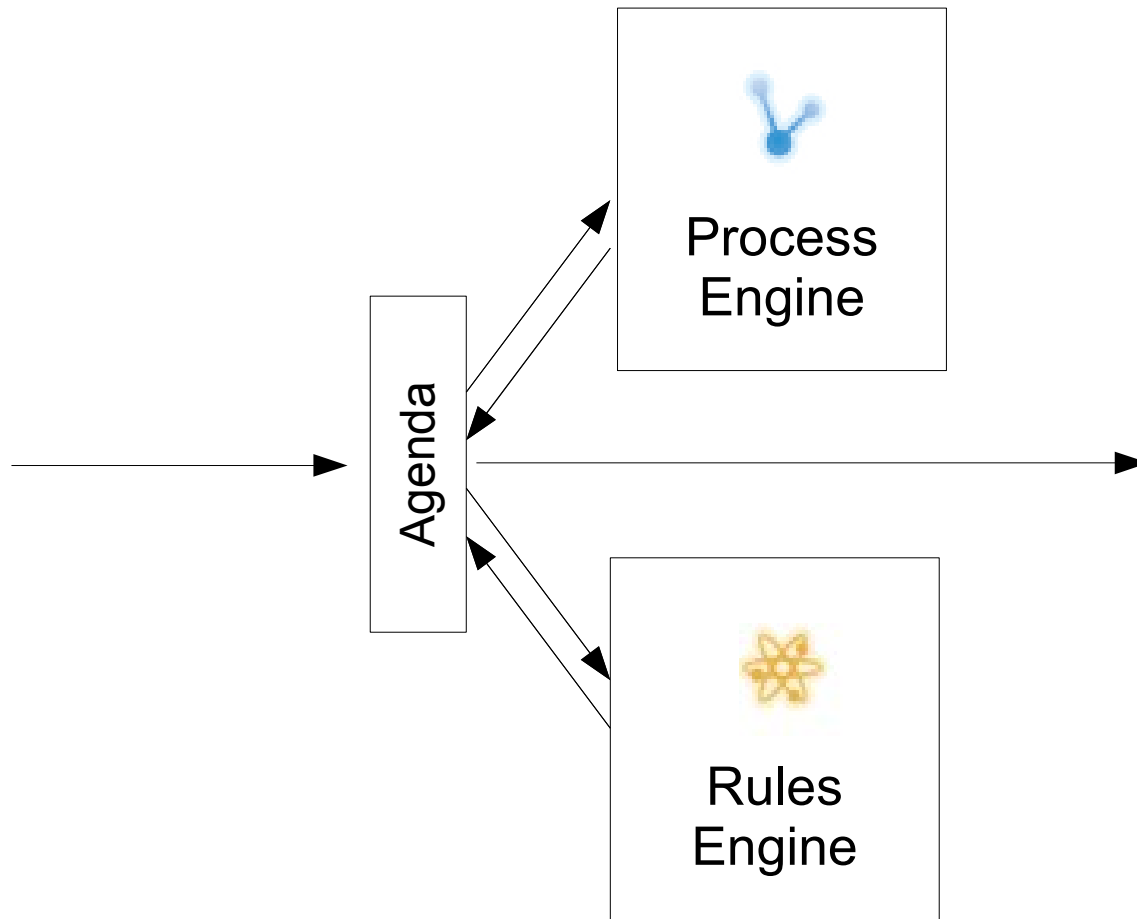
- What if you have a lot of business logic ?



Inversion of Control

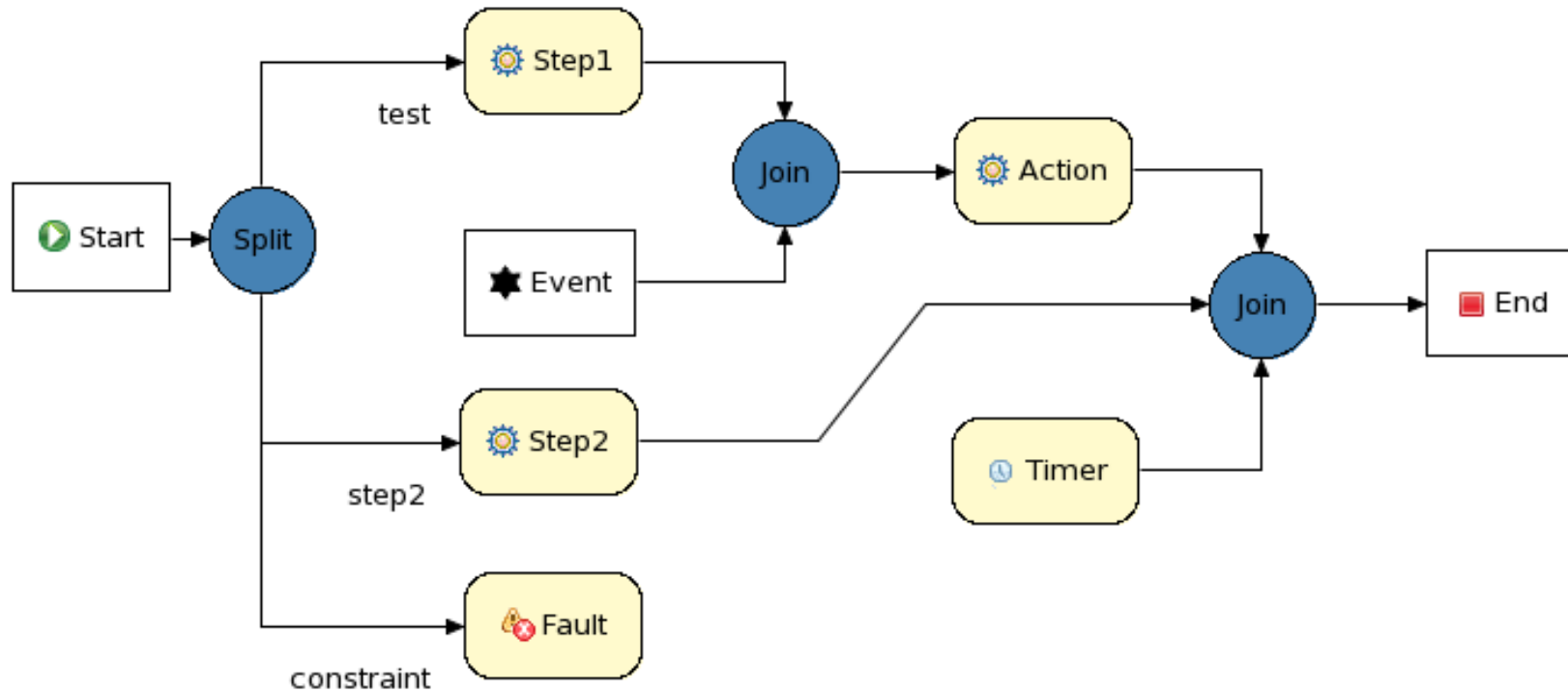


Inversion of Control



Drools | Other Nodes

- 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
 - Log



Drools | Audit View

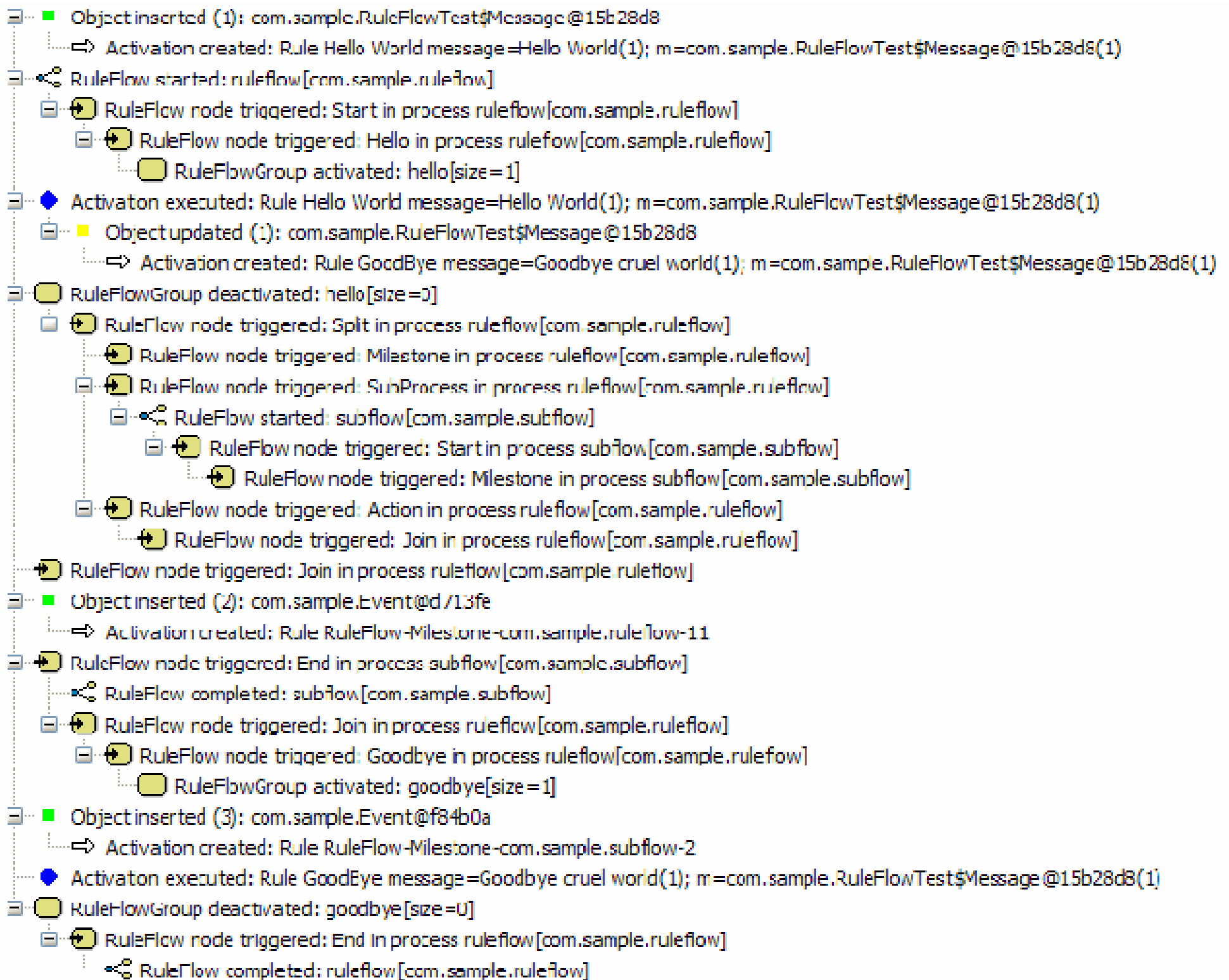
The image shows two side-by-side windows from the Drools IDE. The left window, titled "Audit View", displays a sequence of events in a tree-like structure. The right window, titled "Agenda View", displays the current state of the agenda.

Audit View:

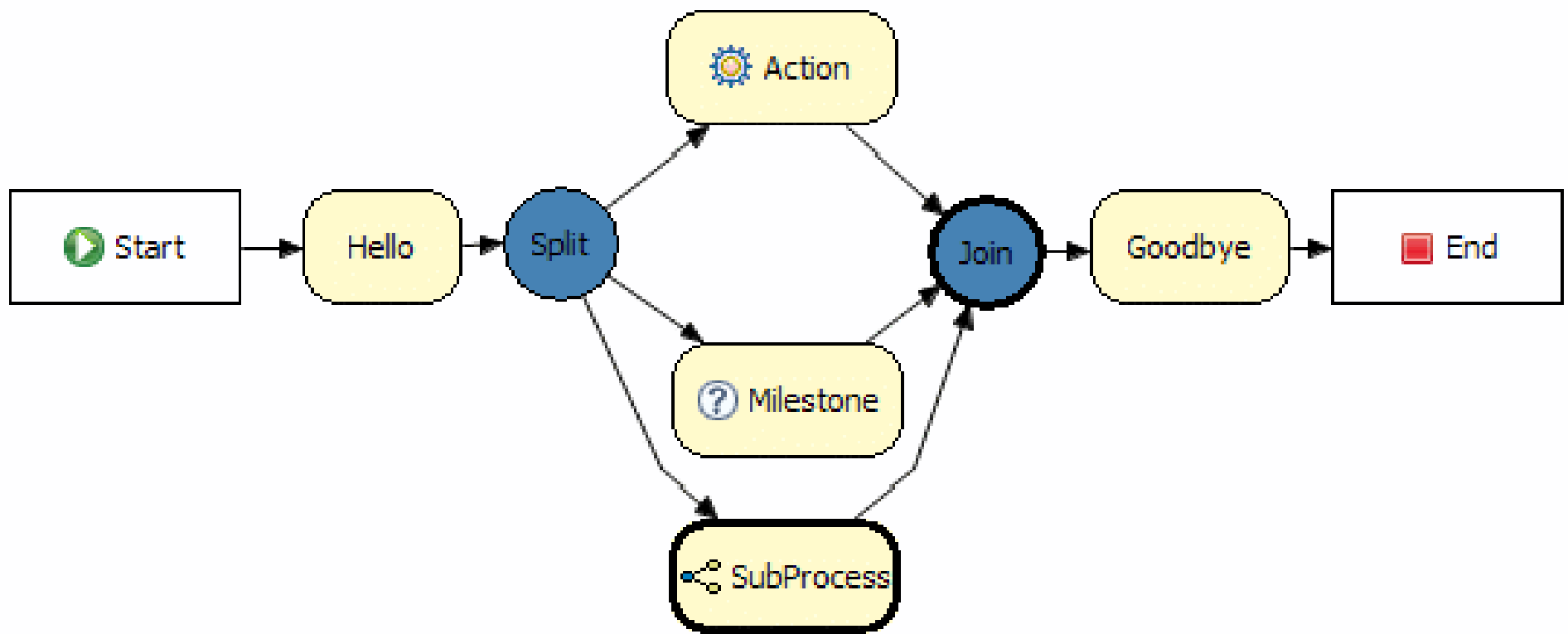
- Object asserted (1): A[NOTRUN]
 - ⇒ Activation created: Rule Bootstrap a=A[NOTRUN](1)
- Object asserted (2): B[NOTRUN]
- Object asserted (3): C[NOTRUN]
- Object asserted (4): D[NOTRUN]
- Activation executed: Rule Bootstrap a=A[NOTRUN](1)
 - Object modified (1): A[FINISHED]
 - ⇒ Activation created: Rule A to B b=B[NOTRUN](2)
- Activation executed: Rule A to B b=B[NOTRUN](2)
 - Object modified (2): B[FINISHED]
 - ⇒ Activation created: Rule B to C c=C[NOTRUN](3)
 - ⇒ Activation created: Rule B to D d=D[NOTRUN](4)
- Activation executed: Rule B to C c=C[NOTRUN](3)
 - Object modified (3): C[FINISHED]
- Activation executed: Rule B to D d=D[NOTRUN](4)
 - Object modified (4): D[FINISHED]

Agenda View:

- MAIN[focus]= AgendaGroupImpl (id=1259)
 - [0]= AgendaItem (id=1262)
 - ruleName= "B to C"
 - c= State (id=1269)
 - [1]= AgendaItem (id=1263)
 - ruleName= "B to D"
 - d= State (id=1270)



Drools | Process Debug



The screenshot displays the Eclipse IDE interface during a debug session. The top status bar shows the file path: `Debug - debug_human_tasks/src/main/rules/validation.drl - Eclipse SDK`. The system tray on the right indicates the date and time: `Tue Oct 21, 02:23`.

The **Debug** window shows the execution stack for the thread `[main] (Suspended (breakpoint at line 7 in Rule_Validation_Rule_0))`. The current line of execution is `Rule_Validation_Rule_0.consequence(KnowledgeHelper, Order, FactHandle) line: 9`.

The **Variables** window shows the current state of variables, with `order` having the value `Order (id=31)`.


The **Code Editor** displays the following code snippet from `validation.drl`:

```
1 package com.sample
2
3 import com.sample.Order
4
5 rule "Validation Rule" // ruleflow-group "Validate"
6   when
7     order: Order( )
8   then
9     System.out.println("Validating order " + order);
10  end
```

The **Process Instance View** shows a process instance `1 = ruleflow[com.sample.ruleflow]` with a flow diagram. The flow starts at `Start`, goes to a `Split` node, which branches into `Task1`, `Task2`, and `Validate`. These three tasks converge at a `Join` node, followed by `Task3`, and finally `End`.

The **Console** window shows the output of the process instance:

```
[1]= RuleFlowProcessInstance (id=4528)
  ▲ id= 1
  ▶ ▲ processName= "ruleflow" (id=4585)
  ▶ ▲ processId= "com.sample.ruleflow" (id=4532)
  ▶ ▲ nodeInstances= Object[] (id=4587)
```

Applications Places System  Tue Oct 21, 02:23

Debug - debug_human_tasks/src/main/rules/validation.drl - Eclipse SDK

File Edit Navigate Search Project Run Window Help

Debug [main] (Suspended (breakpoint at line 7 in Rule_Validation_Rule_0))

- Rule_Validation_Rule_0.consequence(KnowledgeHelper, Order, FactHandle) line: 9
- Rule_Validation_Rule_0ConsequenceInvoker.evaluate(KnowledgeHelper, WorkingMemory) line: 22
- DefaultAgenda.fireActivation(Activation) line: 897
- DefaultAgenda.fireNextItem(AgendaFilter) line: 859
- DefaultAgenda.fireAllRules(AgendaFilter, int) line: 999
- ReteooStatefulSession(AbstractWorkingMemory).fireAllRules(AgendaFilter, int) line: 609
- ReteooStatefulSession(AbstractWorkingMemory).fireAllRules() line: 576
- RuleFlowTest.main(String[]) line: 56
- Thread [Thread-3] (Running)

Variables

Name	Value
order	Order (id=31)

```

1 package com.sample
2
3 import com.sample.Order
4
5 rule "Validation Rule" // ruleflow-group "Validate"
6   when
7     order: Order( )
8   then
9     System.out.println("Validating order " + order);
10  end
    
```

Process Instance View

- Activation created: Rule Validation Rule order=Order O-1234(1)
- RuleFlow started: ruleflow[com.sample.ruleflow]
 - RuleFlow node triggered: Start in process ruleflow[com.sample.ruleflow]
 - RuleFlow node triggered: Split in process ruleflow[com.sample.ruleflow]
 - RuleFlow node triggered: Task1 in process ruleflow[com.sample.ruleflow]
 - RuleFlow node triggered: Validate in process ruleflow[com.sample.ruleflow]
 - RuleFlowGroup activated: Validate[size=0]
 - RuleFlow node triggered: Task2 in process ruleflow[com.sample.ruleflow]
 - RuleFlowGroup deactivated: Validate[size=0]
 - RuleFlow node triggered: Join in process ruleflow[com.sample.ruleflow]
- Activation executed: Rule Validation Rule order=Order O-1234(1)

Console

Userid: Darth Vader Refresh

Name	Status	Owner	Created On	Comment
Task1	InProgress	Darth Vader	Oct 21, 2008 2:08:25 AM	This is some task ...
Task2	Reserved	Darth Vader	Oct 21, 2008 2:08:25 AM	This is some task ...

Claim Start Stop Release Suspend Resume Skip Complete Fail

Task View

Java - drools-eclipse-task/src/main/java/org/drools... Debug - debug_human_tasks/src/main/rules/valid... Buddy List Cindy Casteels

Drools | Pluggable Work Items

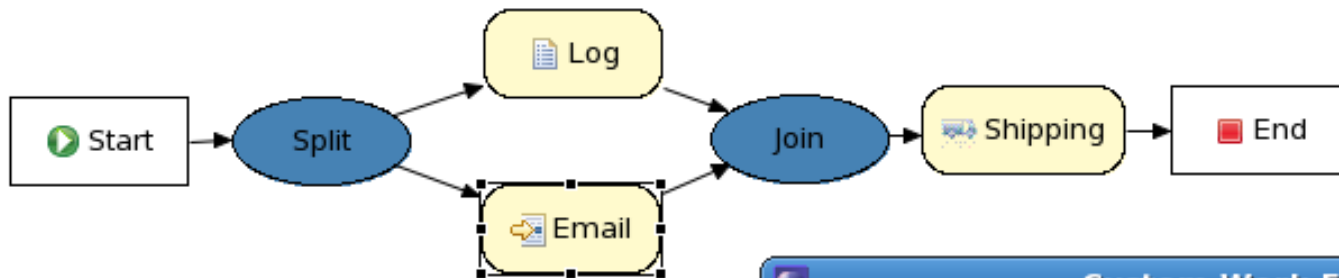
Select
Marquee
Connection Creation

Components

- Start
- End
- RuleFlowGr...
- Split
- Join
- Event Wait
- SubFlow
- Action
- Timer
- Fault
- Event
- Human Task
- Composite
- For Each

Work Items

- Email
- Shipping
- Log



Custom Work Editor

Header Body

Recipients

Email Address

you@mail.com

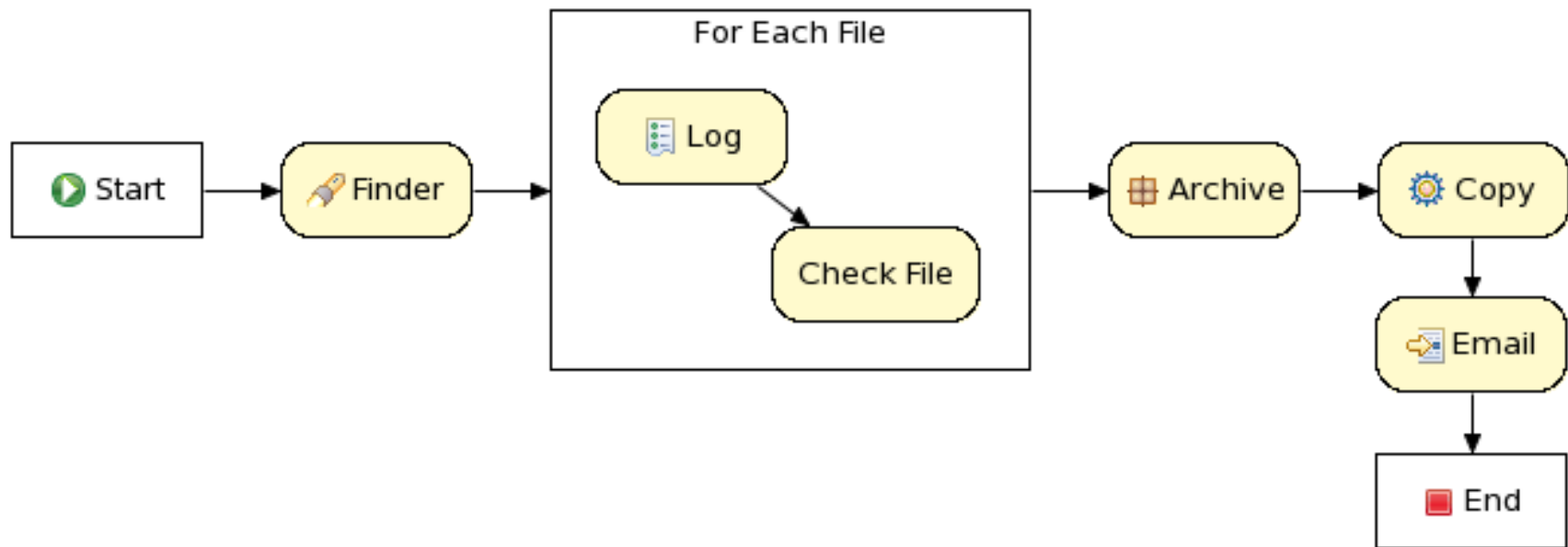
Add Remove

From me@mail.com

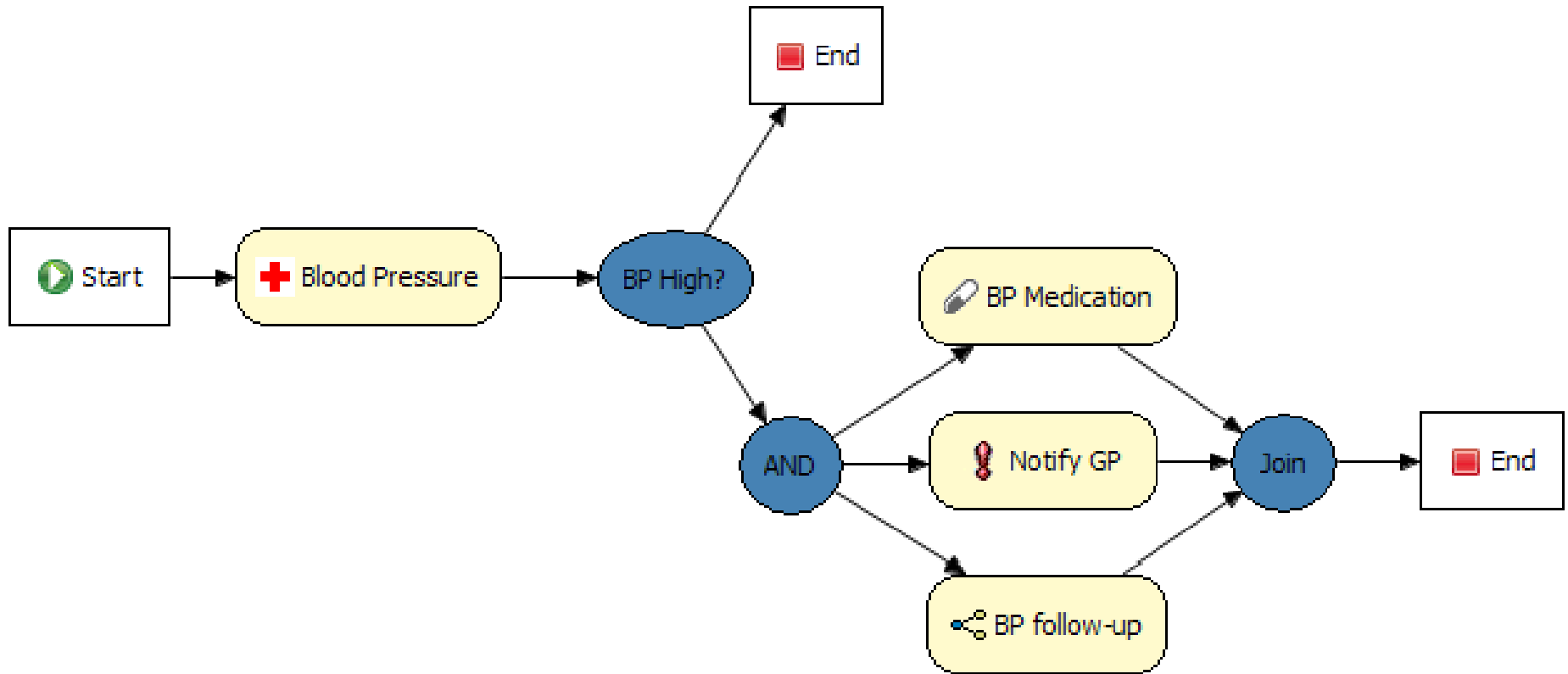
OK Cancel

Drools | Pluggable Work Items

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



Drools | Domain-specific processes



```
<?xml version="1.0" encoding="UTF-8"? >
<process xmlns="http://drools.org/drools-4.0/process"
  xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
  xs:schemaLocation="http://drools.org/drools-4.0/process drools-processes-4.0.xsd"
  type="RuleFlow" name="ruleflow" id="com.sample.ruleflow" package-
  name="com.sample" >

<header>
</header>

<nodes>
  <start id="1" name="Start" x="16" y="16" />
  <actionNode id="2" name="Hello" x="128" y="16" >
    <action type="expression" dialect="mvel" >System.out.println("Hello World");</action>
  </actionNode>
  <end id="3" name="End" x="240" y="16" />
</nodes>

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

</process>
```

```
<process xmlns="http://drools.org/drools-4.0/process"
  xmlns:mysdl="http://domain/org/mysdl"
  xmlns:xs="http://www.w3.org/2001/XMLSchema-instance"
  xs:schemaLocation="http://drools.org/drools-4.0/process drools-processes-4.0.xsd"
  name="process name" id="process name" package-name="org.domain" >
  <nodes>
    <start name="start node" />

    <action name="action node" dialect="java">
      list.add( "action node was here" );
    </action>

    <mysdl:logger name="test logger" type="warn">
      This is my message
    </mysdl:logger>

    <end name="end node" />
  </nodes>

  <connections>
    <connection from="start node" to="action node" />
    <connection from="action node" to="test logger" />
    <connection from="test logger" to="end node" />
  </connections>

</process>
```

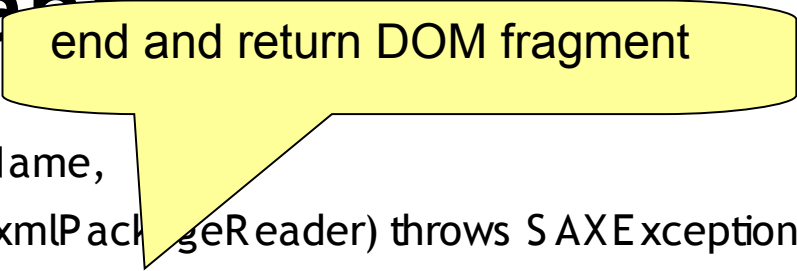
Drools | ePDL Semantic Module Handler

```
SemanticModule module = new DefaultSemanticModule( "http://domain/org/mydsl" );  
module.addHandler( "logger", new LoggerHandler() );
```



Starts Building DOM fragment

```
public Object start(String uri,
                    Attributes attrs,
                    ExtensibleXmlParser xmlParser,
                    ExtensibleXmlPackageReader xmlPackageReader) throws SAXException {
    xmlPackageReader.startConfiguration( localName, attrs );
    RuleFlowProcessImpl process = ( RuleFlowProcessImpl) xmlPackageReader.getParent();
    ActionNodeImpl actionNode = new ActionNodeImpl();
    actionNode.setName( attrs.getValue( "name" ) );
    process.addNode( actionNode );
    ((ProcessBuildData)xmlPackageReader.getData()).addNode( actionNode );
    return actionNode;
}
```

end and return DOM fragment

```
public Object end(String uri, String localName,  
                ExtensibleXmlParser xmlPackageReader) throws SAXException {  
    Configuration config = xmlPackageReader.endConfiguration();  
    RuleFlowProcessImpl process = ( RuleFlowProcessImpl ) xmlPackageReader.getParent();  
    ActionNodeImpl actionNode = ( ActionNodeImpl ) xmlPackageReader.getCurrent();  
    actionNode.setAction( "logger.Warn(" + config.getText() + ")" );  
    return actionNode;  
}
```



- **Dave Bowman:** All right, HAL; I'll go in through the emergency airlock.
- **HAL:** Without your space helmet, Dave, you're going to find that rather difficult.
- **Dave Bowman:** HAL, I won't argue with you anymore! Open the doors!
- **HAL:** Dave, this conversation can serve no purpose anymore. Goodbye.

Joshua: Greetings, Professor Falken.

Stephen Falken: Hello, Joshua.

Joshua: A strange game. The only winning move is not to play. How about a nice game of chess?