FOLLOW US:

TWITTER.COM/REDHATSUMMIT

TWEET ABOUT US:

ADD #SUMMIT AND/OR #JBOSSWORLD TO THE END OF YOUR EVENT-RELATED TWEET

JBoss WORLD
CHICAGO 2009

JBoss Drools: State of the Union

Edson Tirelli
Sr Software Engineer, Red Hat

September 2009

presented by

Agenda

- Business Knowledge
- Drools Expert
- Drools Flow
- Drools Fusion
- Drools Guvnor
- Demo

Business Knowledge

What is **Business Knowledge**?

Is legislation Business Knowledge?

"**Dept Store:** All products sold in California, are due 8.25% of state sales tax."

Legislation ✓ YES

Are company policies Business Knowledge?

"**Telco:** any new subscribers in September/2009 are entitled to a 50% discount on their subscription fees for 60 days."

Company policy ✓ YES

Is Business Intelligence Business Knowledge?

"**Algorithmic Trading:** when a given asset falls more than 5% in 2 minutes, after a government economic announcement, reevaluate our position and adjust the exposure to that asset."

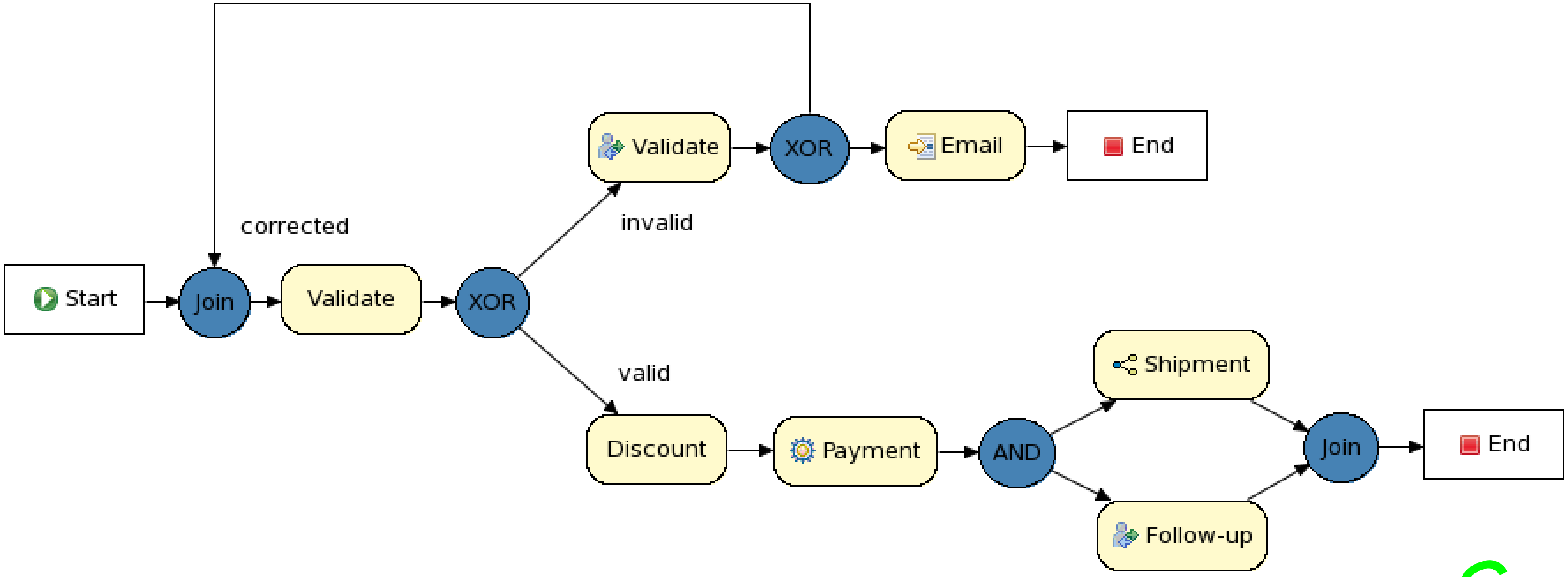
Business Intelligence ✓ YES

Is Operational Data Business Knowledge?

Country of Residence	Age	Gender	Risk Factor
US	< 21	Male	1.00
		Female	0.80
	>= 21	Male	1.20
		Female	1.30
Canada	< 18	Male	0.90
		Female	1.00
	>= 18	Male	1.20
		Female	1.10

Operational Data ✓ YES

Are Business Processes Business Knowledge?



Business process ✓ YES

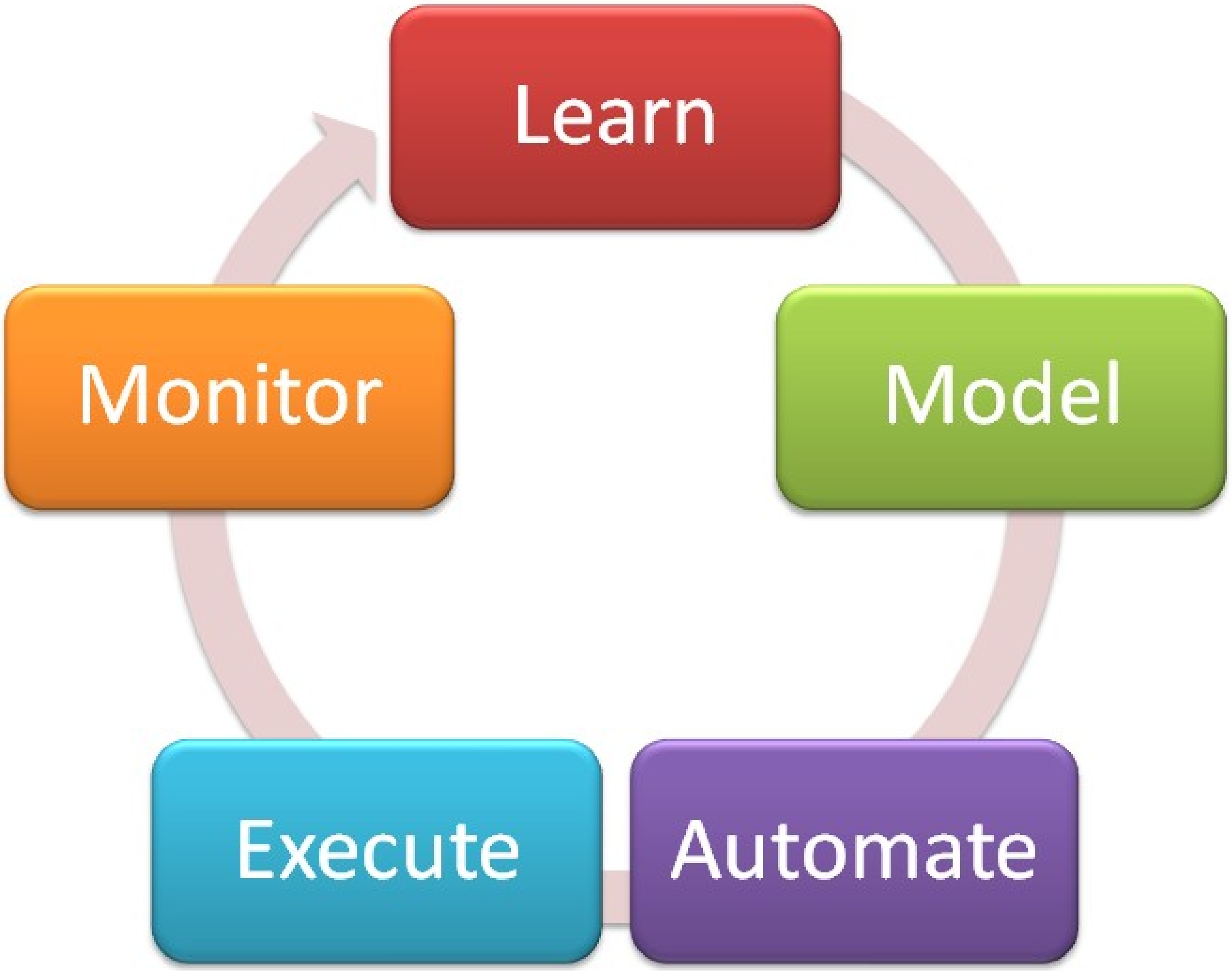
Business Knowledge is:

- Using a loose definition:
  - **Business Logic:** the understanding of **what** to do and **when** to do things in order to run a business.

+

  - **Operational Logic:** the understanding of **how** to do things in order to run a business.

Business Logic Management



Learn

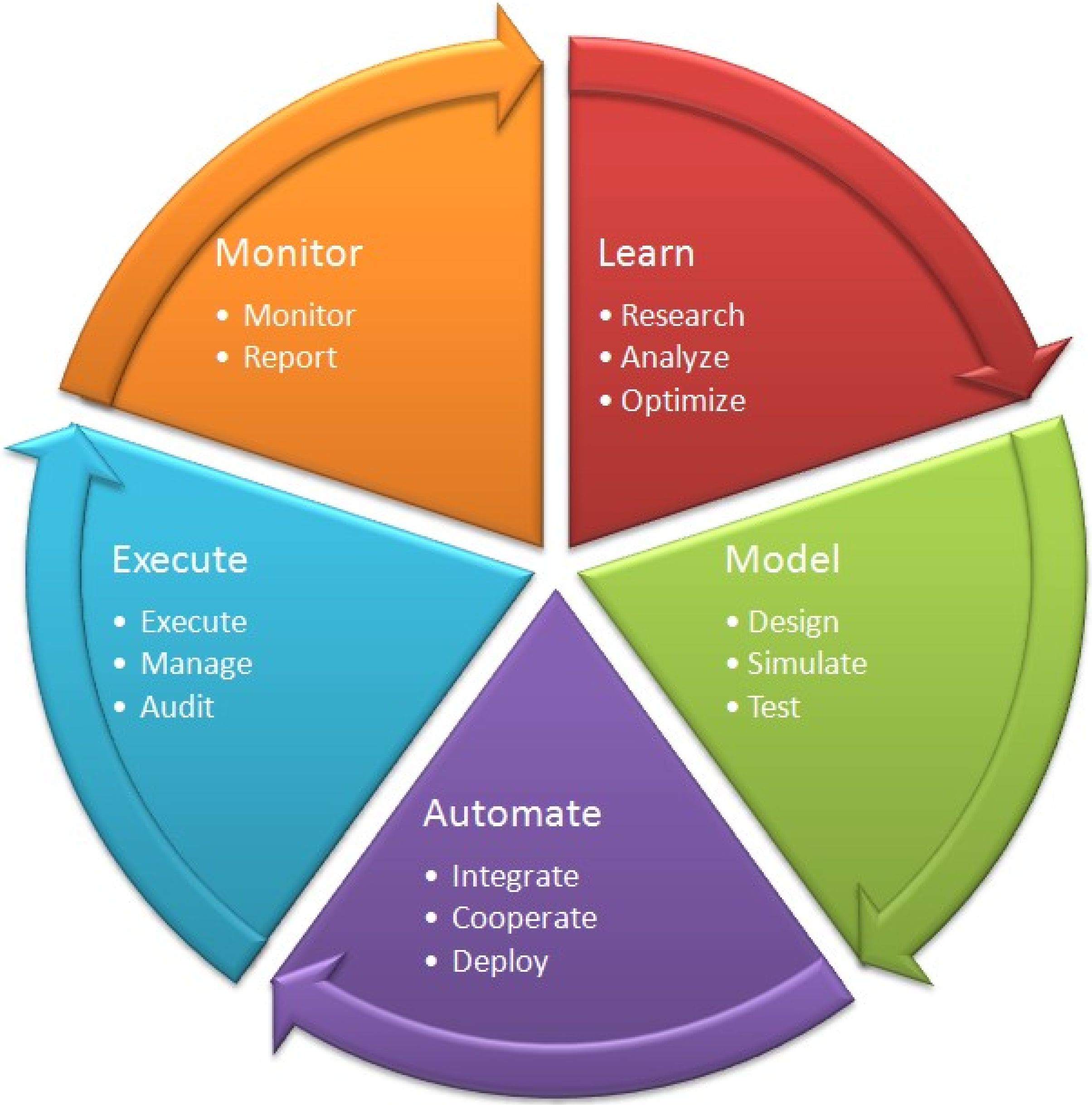
Model

Automate

Execute

Monitor

Business Logic Management



Learn

- Research
- Analyze
- Optimize

Model

- Design
- Simulate
- Test

Automate

- Integrate
- Cooperate
- Deploy

Execute

- Execute
- Manage
- Audit

Monitor

- Monitor
- Report

Drools Vision

Drools
Business Logic integration Platform
Guvnor
Expert
Fusion
Flow

“A common platform to manage the business logic of the enterprise.”

Drools Business Logic Integration Platform

Drools Expert

- State of the art Rules Engine
  - Full Forward Chaining ReteOO algorithm
  - Multiple execution modes
  - Completely dynamic Knowledge Base management
  - Full support for Predicate Logic and First Order Logic
  - Complete set of flow control features
  - POJOs as facts (no mapping/data copy required)

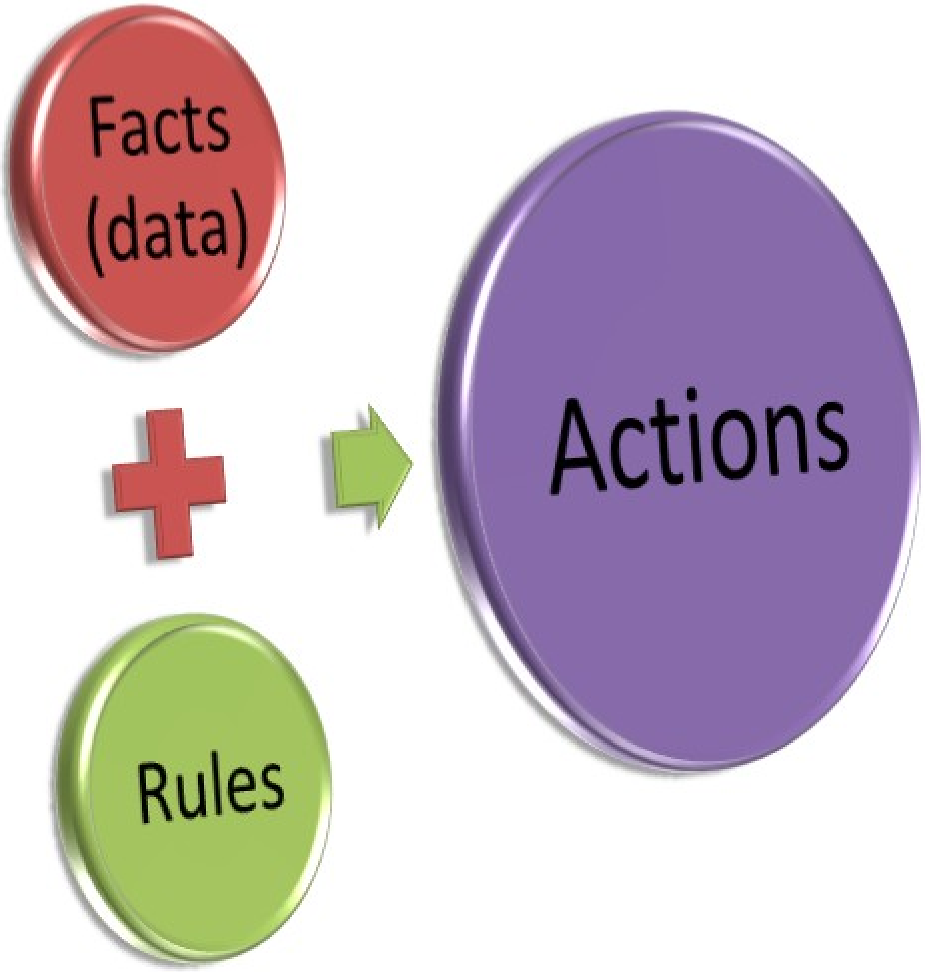
Drools Expert

- Rules format:
  - DRL: Drools Rule Language (native)
  - DSL: Domain Specific Language (user defined)
  - BRL: Business Rule Language (web authoring tools)
  - XML: helps with integration
  - Decision tables: Excel, Open Office, Web

Benefits of a Rules Engine

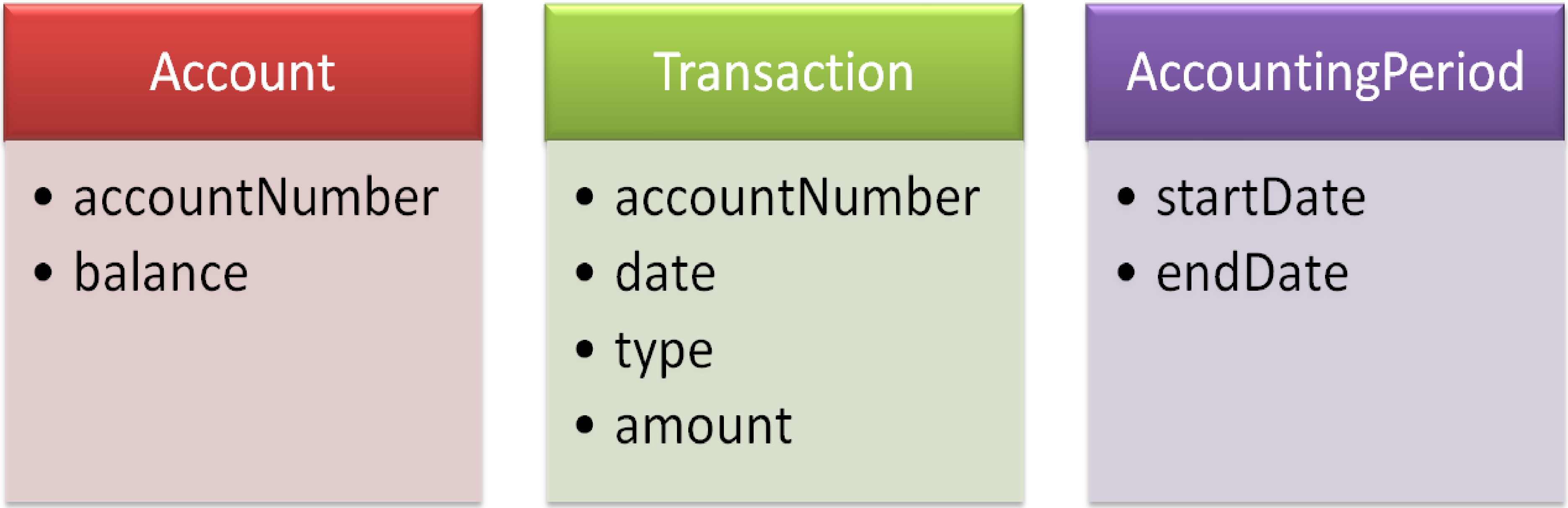
- Decouples application code from business rules
- Enables independent lifecycle management
- Simplifies the development of complex use cases
- Improves business rules maintainability
- Decreases turn-around times for business changes
- Improves performance for complex set of rules
- Closes the gap between Business Users and IT

How a rules engine work?



- Compares all facts against all rules
- Matches trigger rule consequences (actions)
- Actions might cause new rules to match (chaining)

Learn by example: Domain Model



Account

- accountNumber
- balance

Transaction

- accountNumber
- date
- type
- amount

AccountingPeriod

- startDate
- endDate

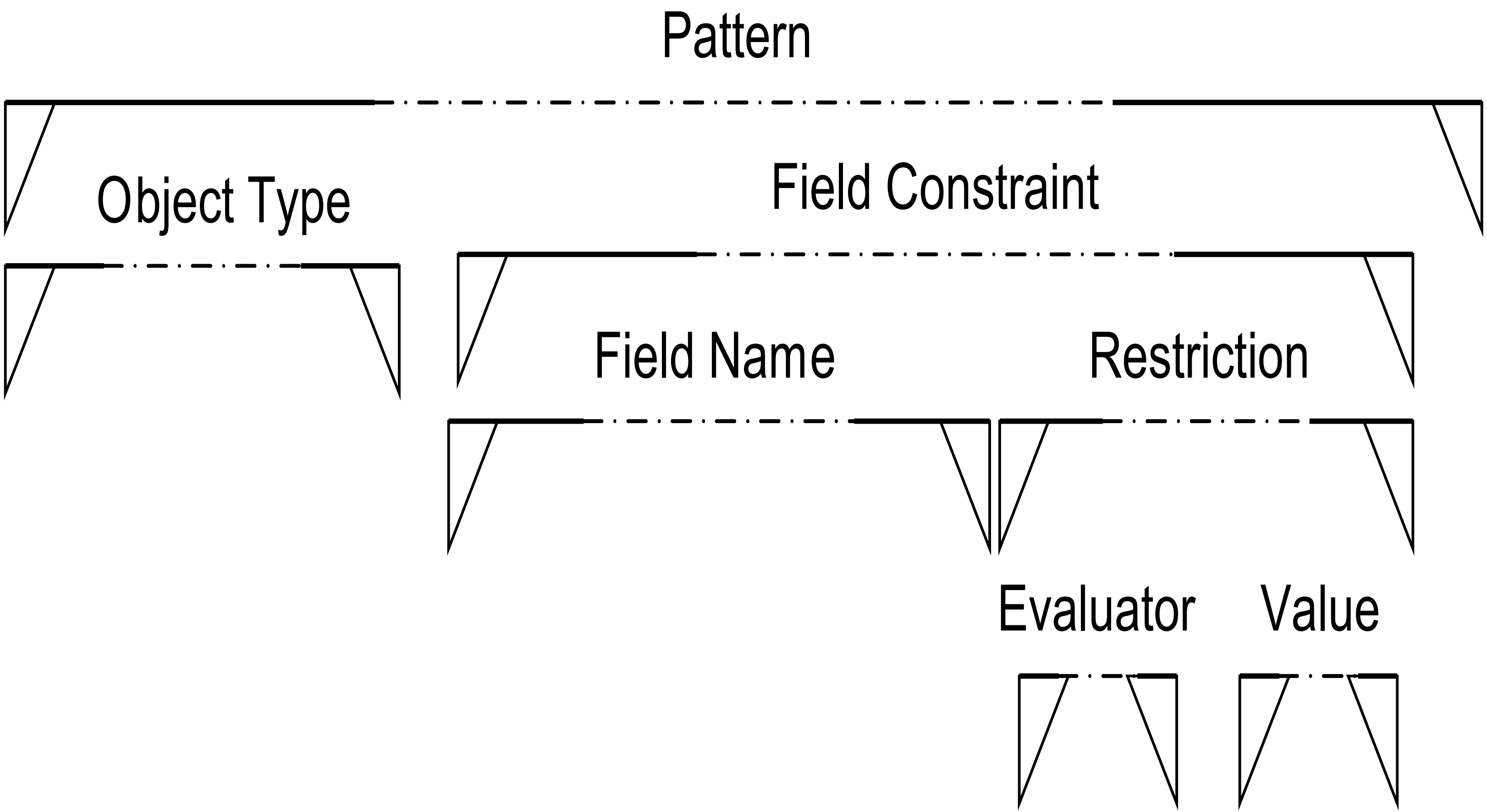
Learn by example: Rule (using DSL)

```
rule “Decrease balance on a debit operation”
when
    There is an account
    There is a transaction whose type is DEBIT
then
    Decrease the account balance by the transaction amount
end
```

Learn by example: Rule (using DRL)

```
rule “Decrease balance on a debit operation”
when
    $account : Account( $actNbr : accountNumber )
    $transaction : Transaction( accountNumber == $actNbr,
                                type == DEBIT )
then
    $account.balance -= $transaction.amount;
end
```

Learn by example: Pattern



Pattern

Object Type

Field Constraint

Field Name

Restriction

Evaluator

Value

Shower(temperature == “hot”)

Learn by example: Patterns

```
CashFlow( type == “credit” )
```

```
$ap : AccountPeriod()
CashFlow( date >= $ap.start )
```

```
$ap : AccountPeriod()
CashFlow( date >= $ap.start && <= $ap.end )
```

```
$ap : AccountPeriod()
CashFlow( type == “credit”,
          date >= $ap.start && <= $ap.end )
```

Learn by example: more Patterns

```
Person( $age : age )
Person( age == ( $age + 1 ) )

Person( $age : age )
Person( eval( age == $age + 1 ) )
```

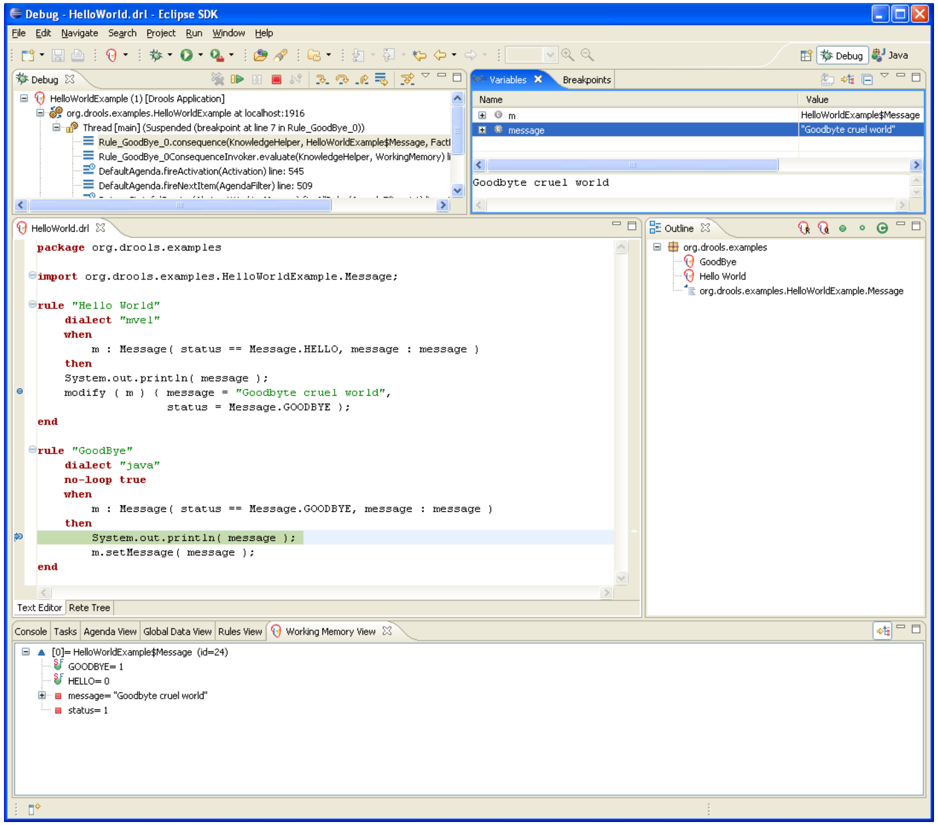
Learn by examples: yet more Patterns

```
Person(age > 30 && < 40 || hair == “black”)

Person(age > 30 && < 40 || hair in (“black”, “brown”) )

Person( (age > 30 && < 40 && hair == “black”)
          ||
        (age > 50 && hair == “grey”) )

Person(pets contain $rover )

Person(pets[’rover’].type == “dog”)
```

Learn by example: Conditional Elements

```
not Bus( color = “red” )

exists Bus( color = “red” )

forall ( $bus : Bus( color == “red” ) )

forall ( $bus : Bus( floors == 2 )
                Bus( this == $bus, color == “red” ) )
```

Drools Expert: IDE



```
HelloWorldExample (1) [Drools Application]
  org.drools.examples.HelloWorldExample at localhost:1916
    Thread [main] (Suspended (breakpoint at line 7 in Rule_GoodBye_0))
      Rule_GoodBye_0.consequence(KnowledgeHelper, HelloWorldExample$Message, FactI
      Rule_GoodBye_0ConsequenceInvoker.evaluate(KnowledgeHelper, WorkingMemory) li
      DefaultAgenda.fireActivation(Activation) line: 545
      DefaultAgenda.fireNextItem(AgendaFilter) line: 509
```

Name	Value
m	HelloWorldExample$Message
message	"Goodbyte cruel world"

```
Goodbyte cruel world
```

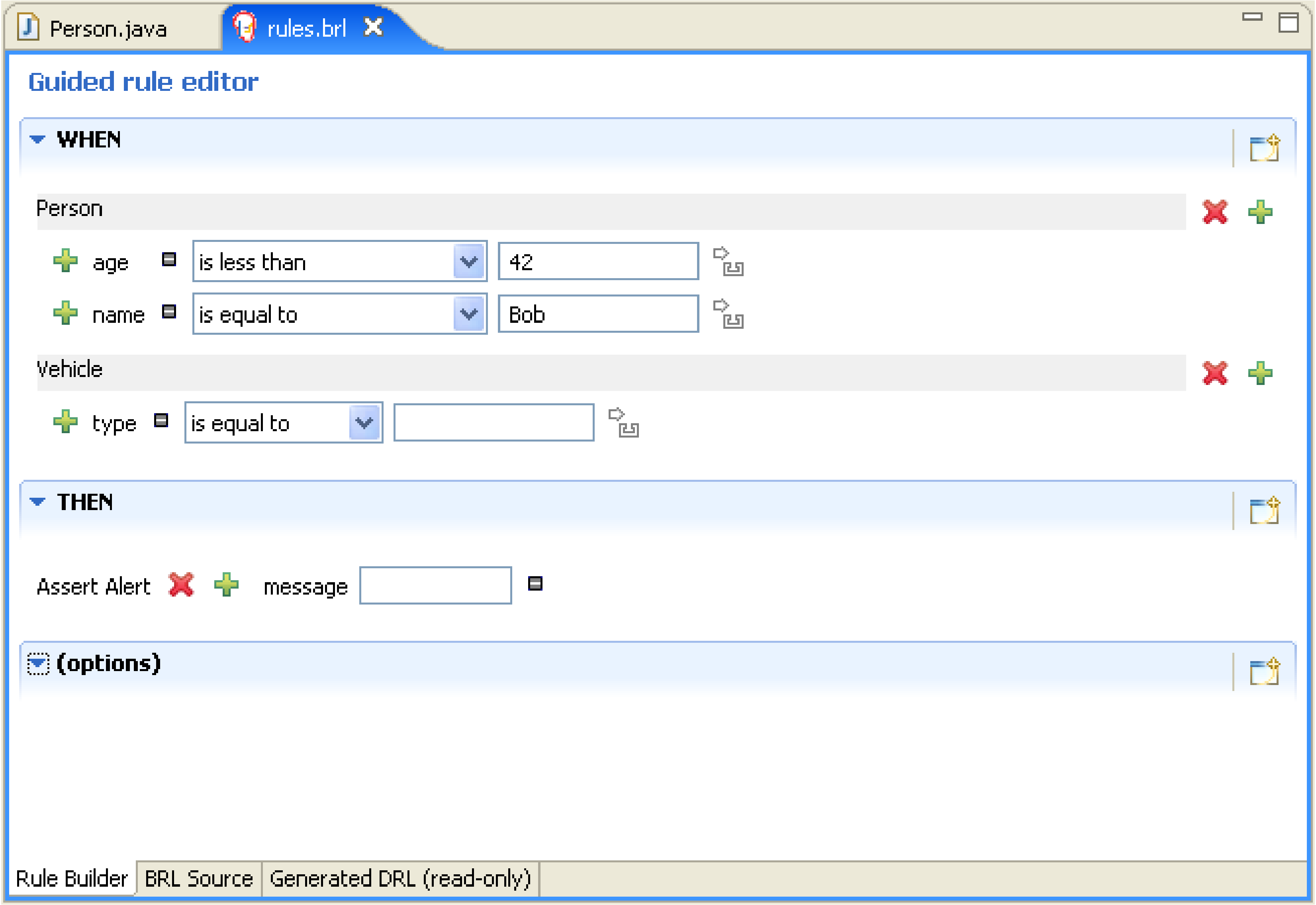
```
package org.drools.examples

import org.drools.examples.HelloWorldExample.Message;

rule "Hello World"
    dialect "mvel"
    when
        m : Message( status == Message.HELLO, message : message )
    then
    System.out.println( message );
    modify ( m ) { message = "Goodbyte cruel world",
                   status = Message.GOODBYE };
end

rule "GoodBye"
    dialect "java"
    no-loop true
    when
        m : Message( status == Message.GOODBYE, message : message )
    then
        System.out.println( message );
        m.setMessage( message );
end
```

```
org.drools.examples
  GoodBye
  Hello World
  org.drools.examples.HelloWorldExample.Message
```

```
[0]= HelloWorldExample$Message  (id=24)
  GOODBYE= 1
  HELLO= 0
  message= "Goodbyte cruel world"
  status= 1
```

- Eclipse-based
- Full feature enabled:
  - Editor
  - Compiler
  - Debugging
  - Audit

Drools Expert: Guided Editor



Person.java | rules.brl

Guided rule editor

WHEN

Person
- age: is less than 42
- name: is equal to Bob

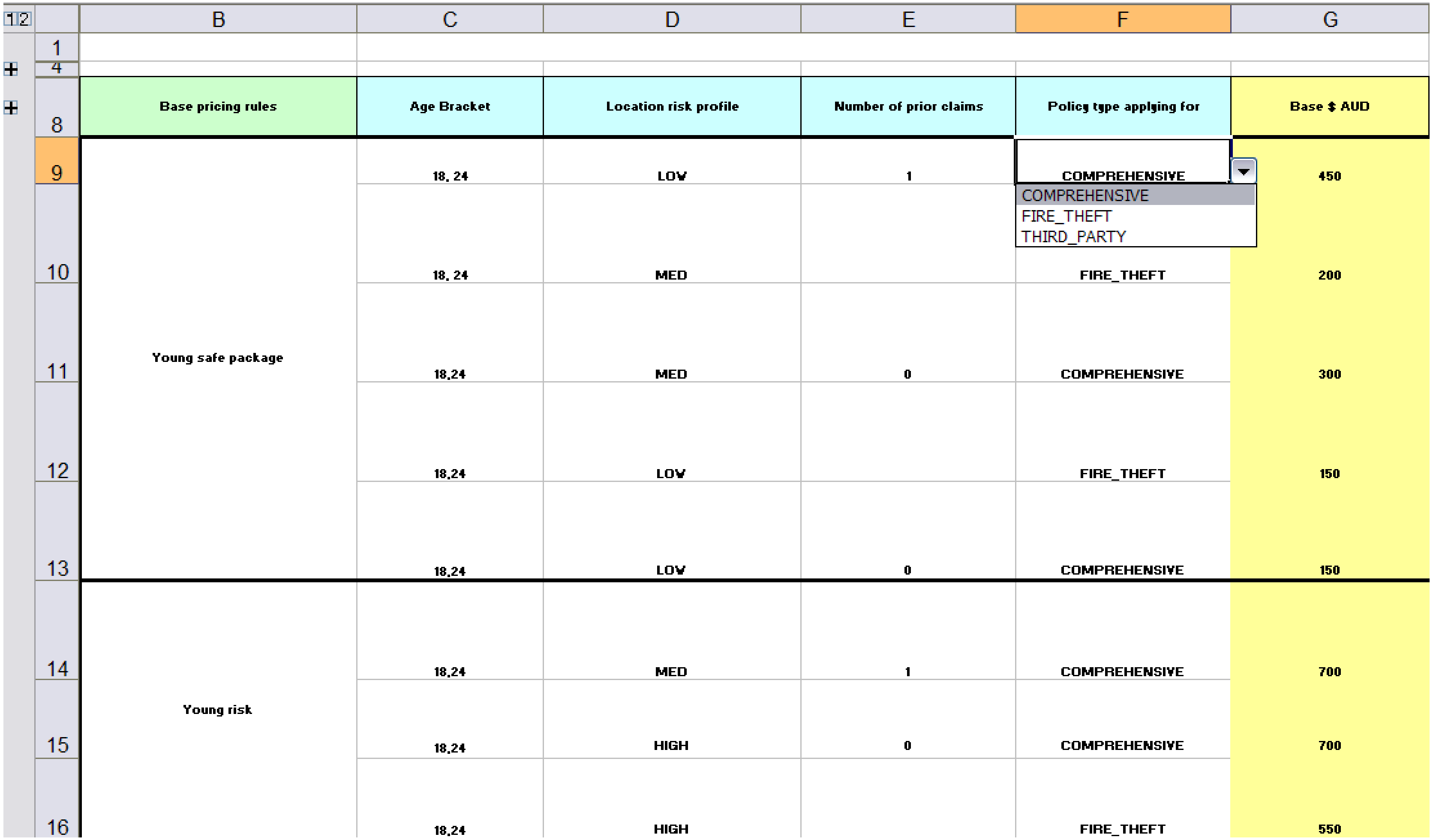
Vehicle
- type: is equal to

THEN

Assert Alert message

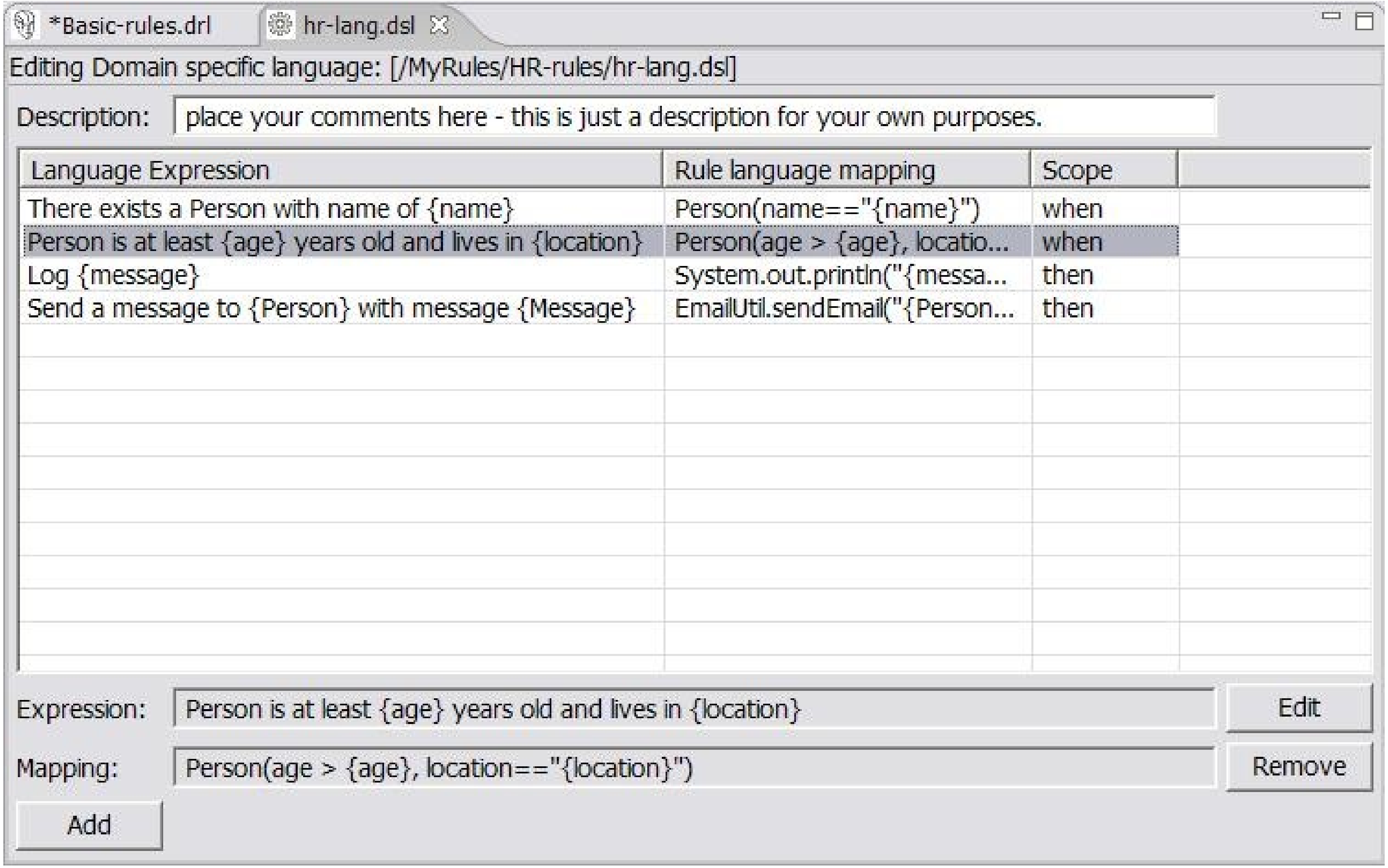
(options)

Drools Expert: Decision Tables



	B	C	D	E	F	G
1						
4						
8	Base pricing rules	Age Bracket	Location risk profile	Number of prior claims	Policy type applying for	Base $ AUD
9	Young safe package	18, 24	LOW	1	COMPREHENSIVE	450
10		18, 24	MED		FIRE_THEFT	200
11		18,24	MED	0	COMPREHENSIVE	300
12		18,24	LOW		FIRE_THEFT	150
13		18,24	LOW	0	COMPREHENSIVE	150
14	Young risk	18,24	MED	1	COMPREHENSIVE	700
15		18,24	HIGH	0	COMPREHENSIVE	700
16		18,24	HIGH		FIRE_THEFT	550

Drools Expert: Domain Specific Languages (DSL)



Editing Domain specific language: [/MyRules/HR-rules/hr-lang.dsl]

Description: place your comments here - this is just a description for your own purposes.

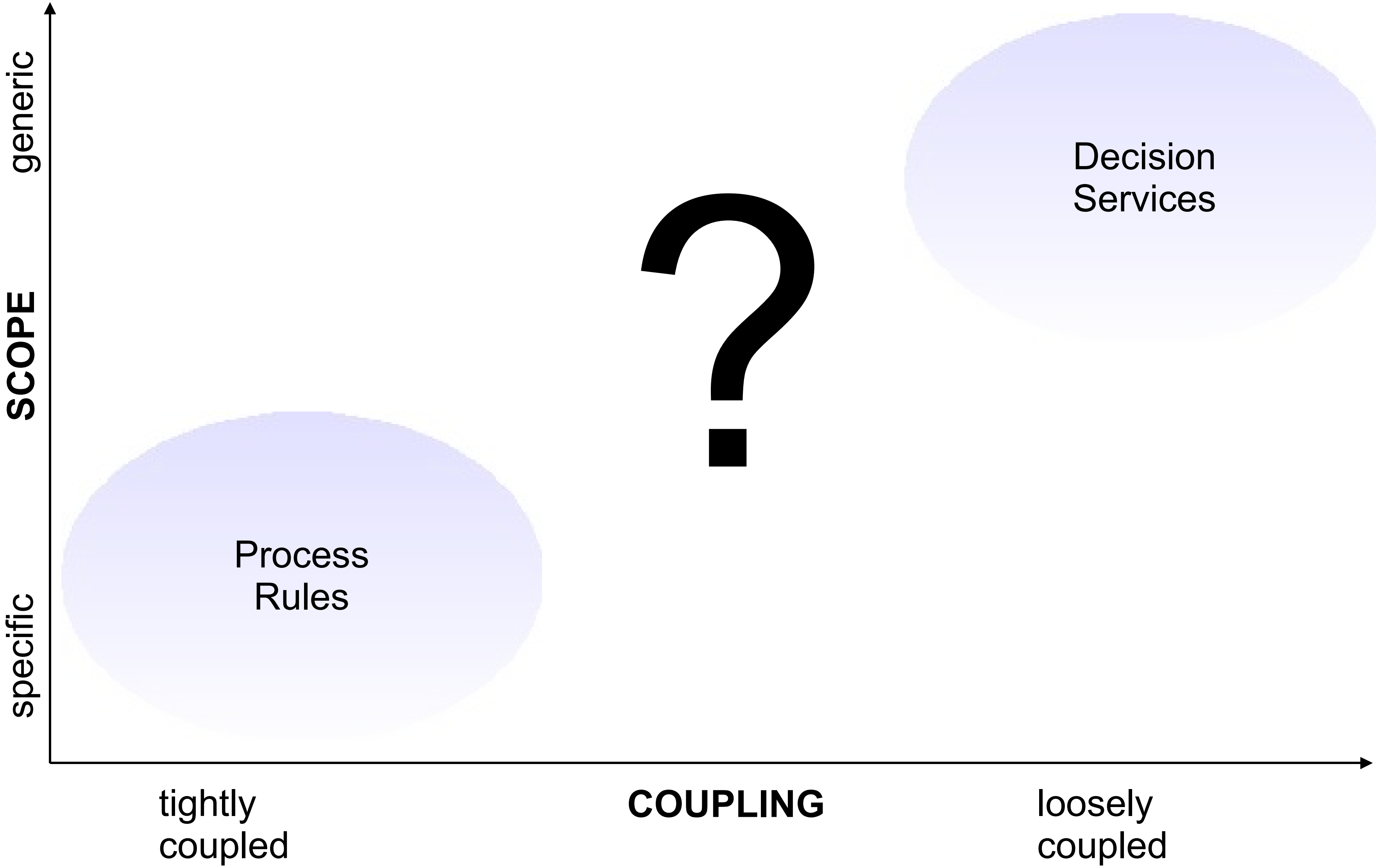
Language Expression	Rule language mapping	Scope	
There exists a Person with name of {name}	Person(name=="{name}")	when	
Person is at least {age} years old and lives in {location}	Person(age > {age}, locatio...	when	
Log {message}	System.out.println("{messa...	then	
Send a message to {Person} with message {Message}	EmailUtil.sendEmail("{Person...	then	

Expression: Person is at least {age} years old and lives in {location}

Mapping: Person(age > {age}, location=="{location}")

Drools Business Logic Integration Platform

Rules and Processes



generic

SCOPE

specific

Decision Services

?

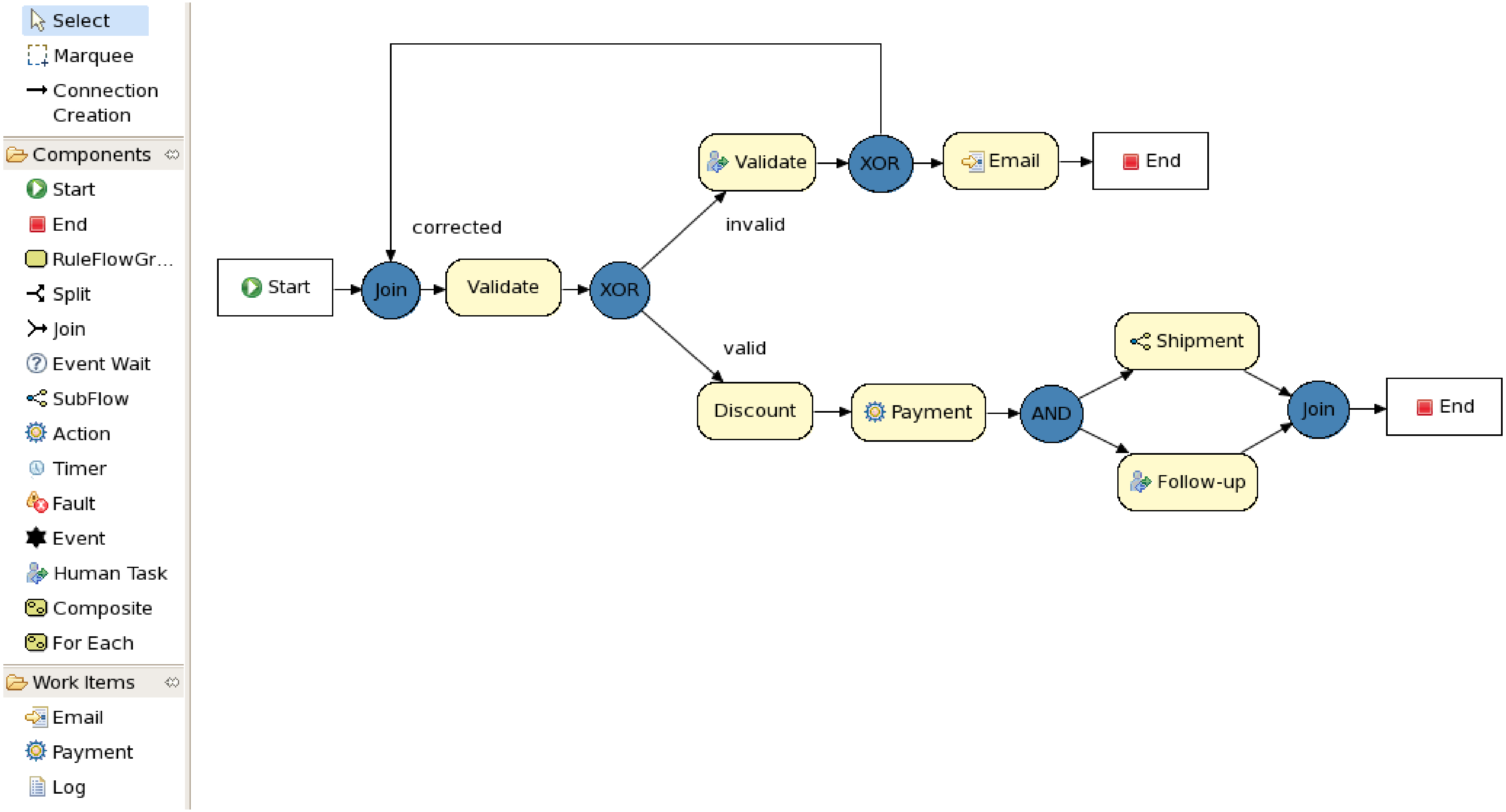
Process Rules

tightly coupled

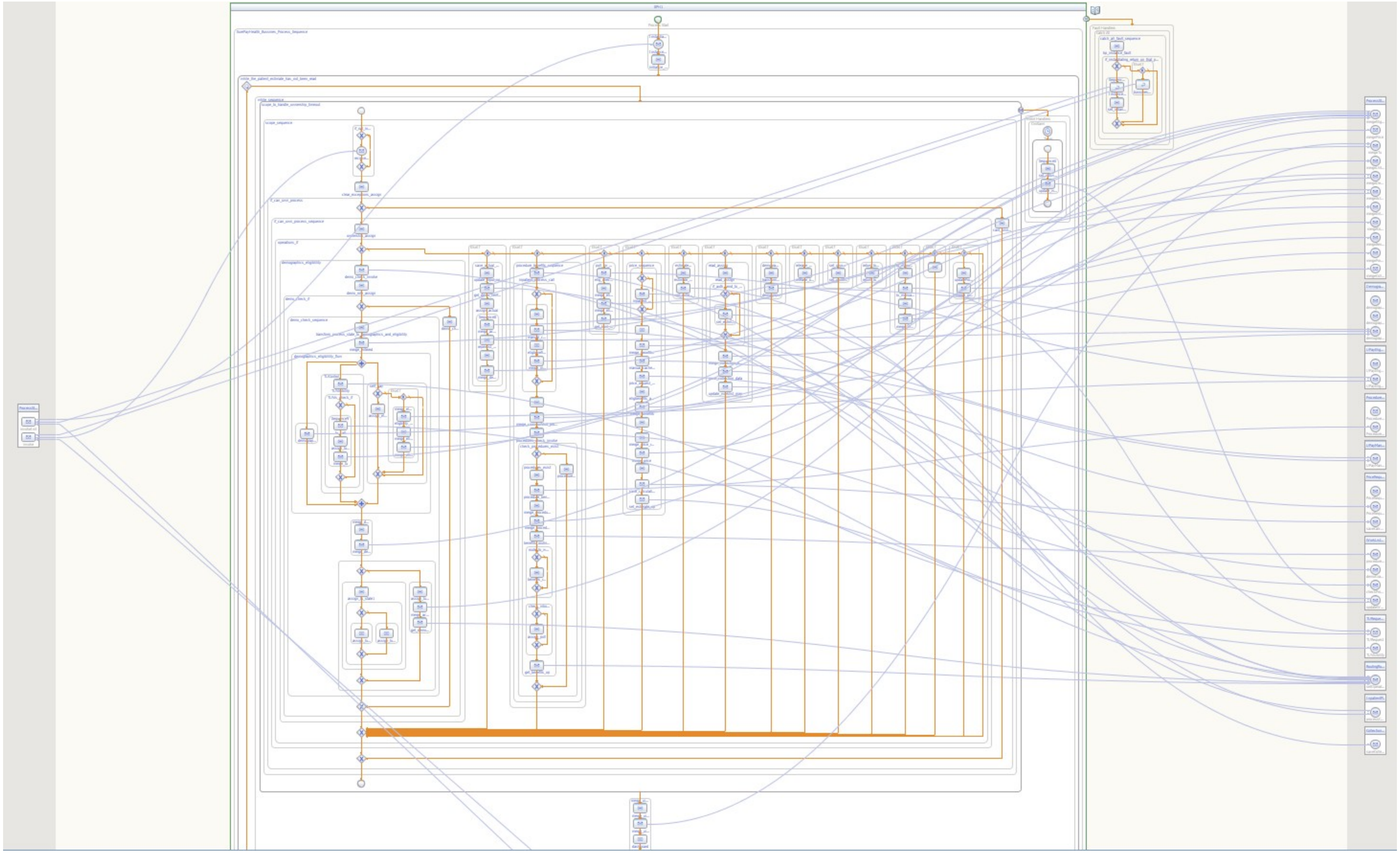
COUPLING

loosely coupled

Drools Flow

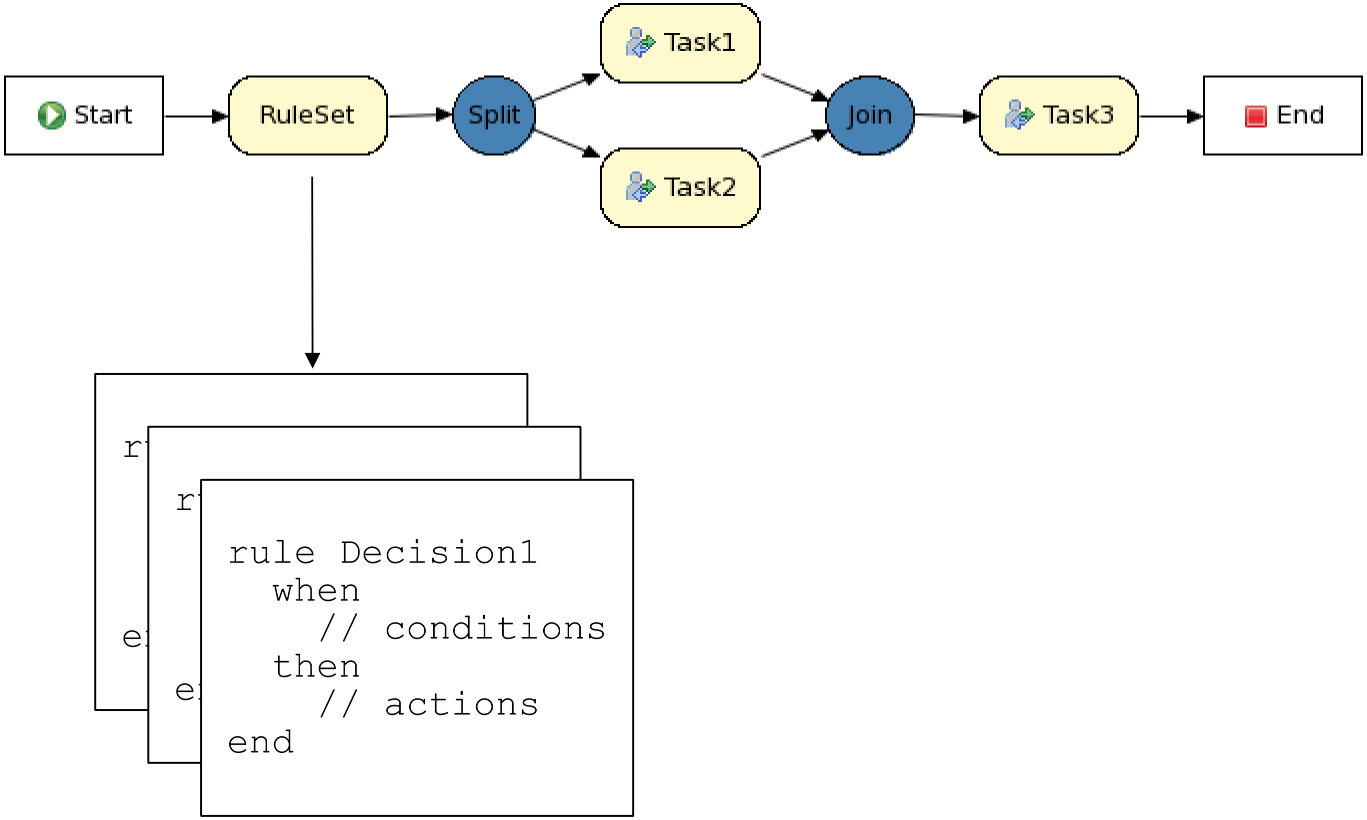

Select
Marquee
Connection Creation
Components
Start
End
RuleFlowGr...
Split
Join
Event Wait
SubFlow
Action
Timer
Fault
Event
Human Task
Composite
For Each
Work Items
Email
Payment
Log
Start
Join
Validate
XOR
corrected
invalid
valid
Validate
XOR
Email
End
Discount
Payment
AND
Shipment
Follow-up
Join
End

BPEL Hell



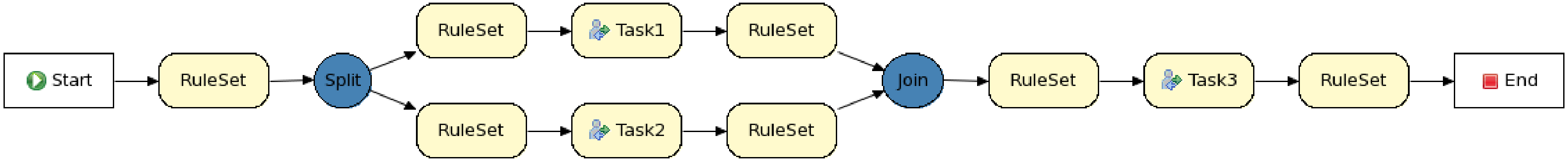
Using Decision Services in a Process

Business decisions are externalized using a decision service

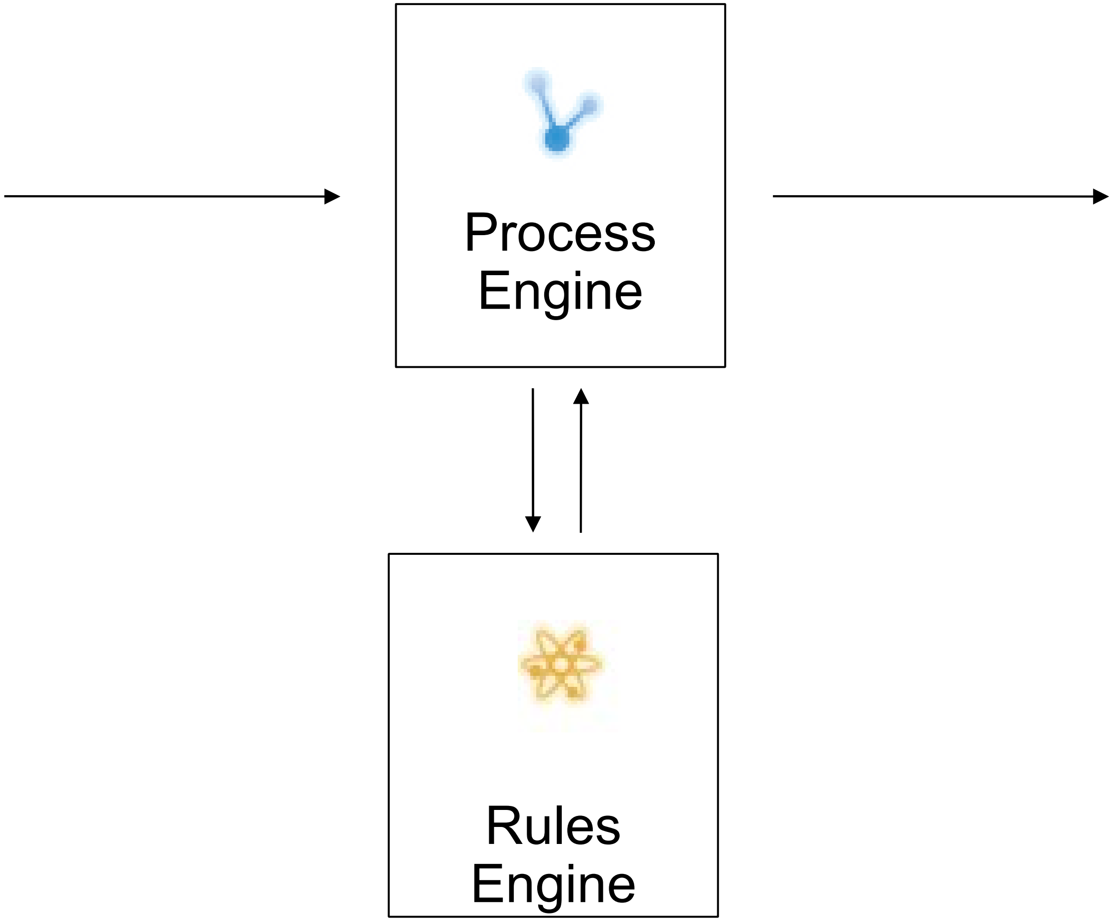


```
rule Decision1
  when
    // conditions
  then
    // actions
end
```

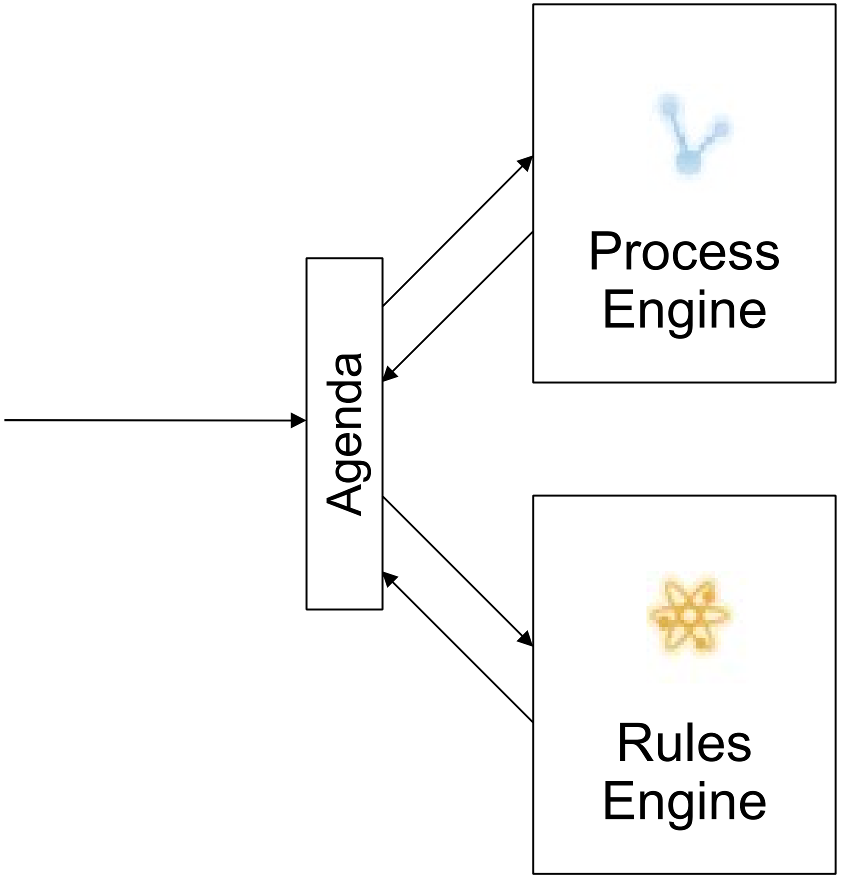
Example


Start
RuleSet
Split
RuleSet
Task1
RuleSet
RuleSet
Task2
RuleSet
Join
RuleSet
Task3
RuleSet
End

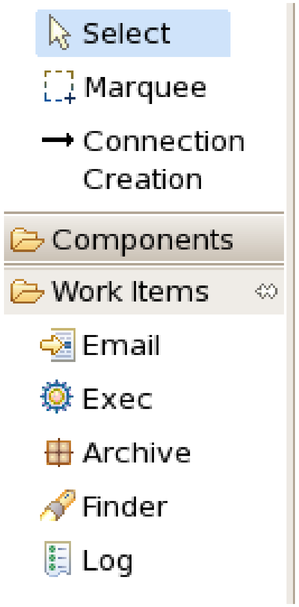
Drools Flow: Usual flow of control

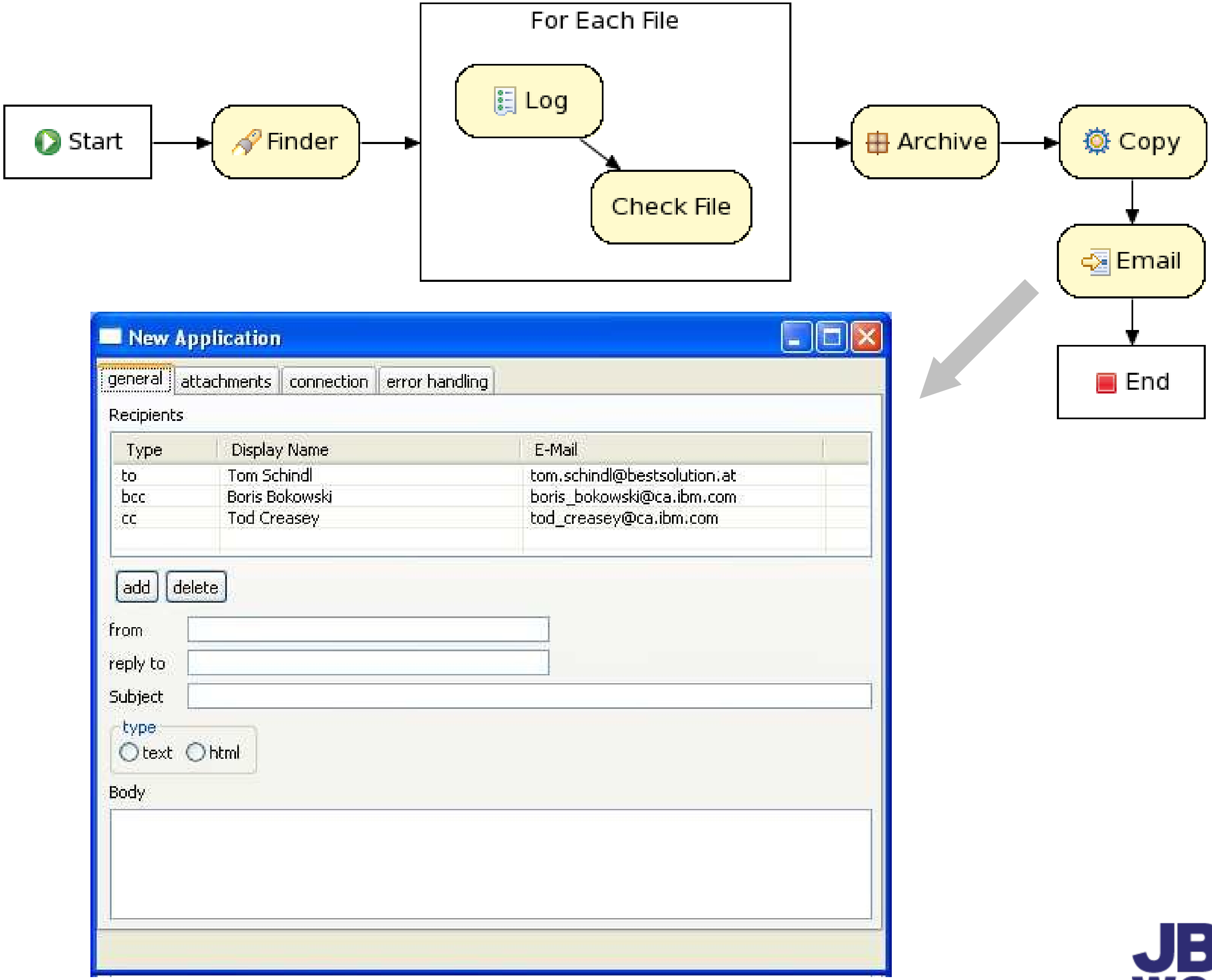


Drools Flow: Inversion of Control



Drools Flow: Domain Specific Processes





New Application

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

from

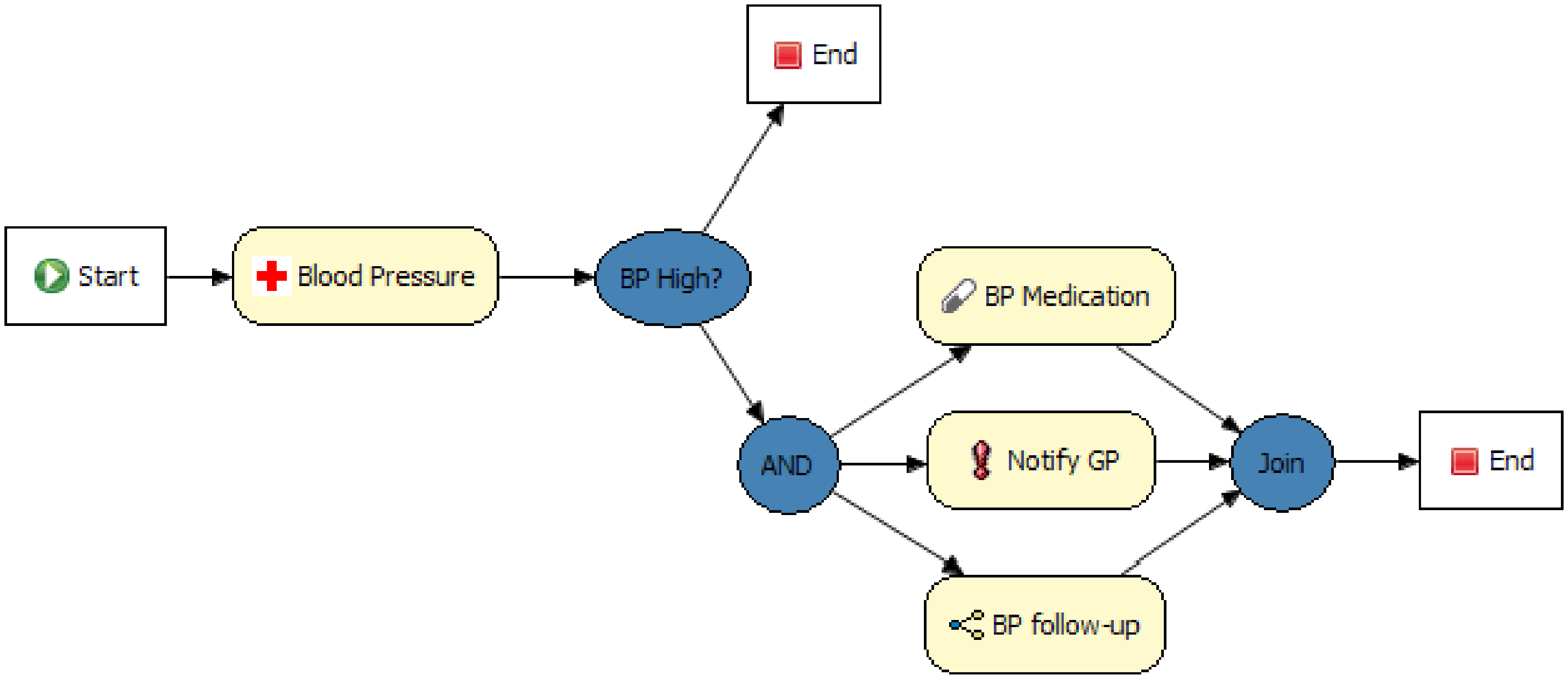
reply to

Subject

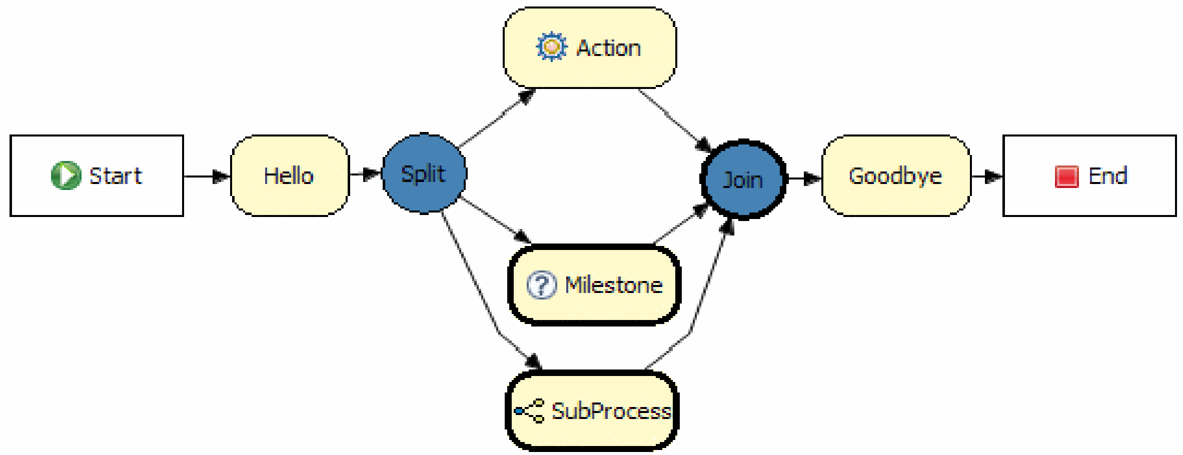
type
○ text ○ html

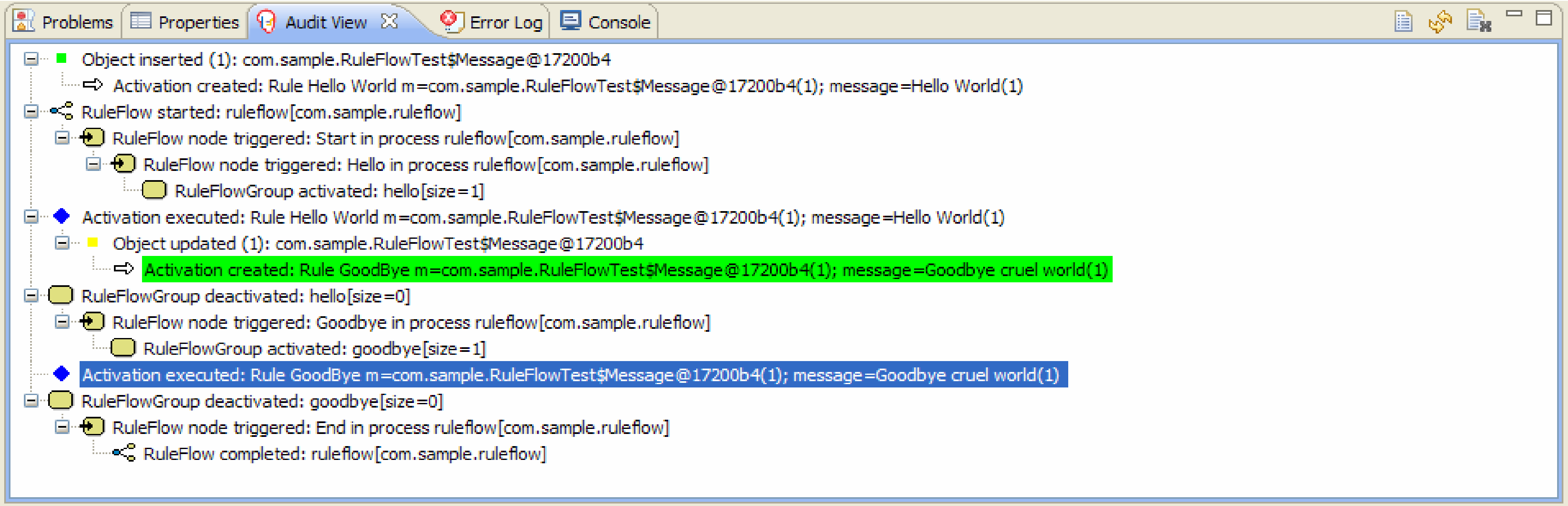
Body

Drools Flow: Domain Specific Processes



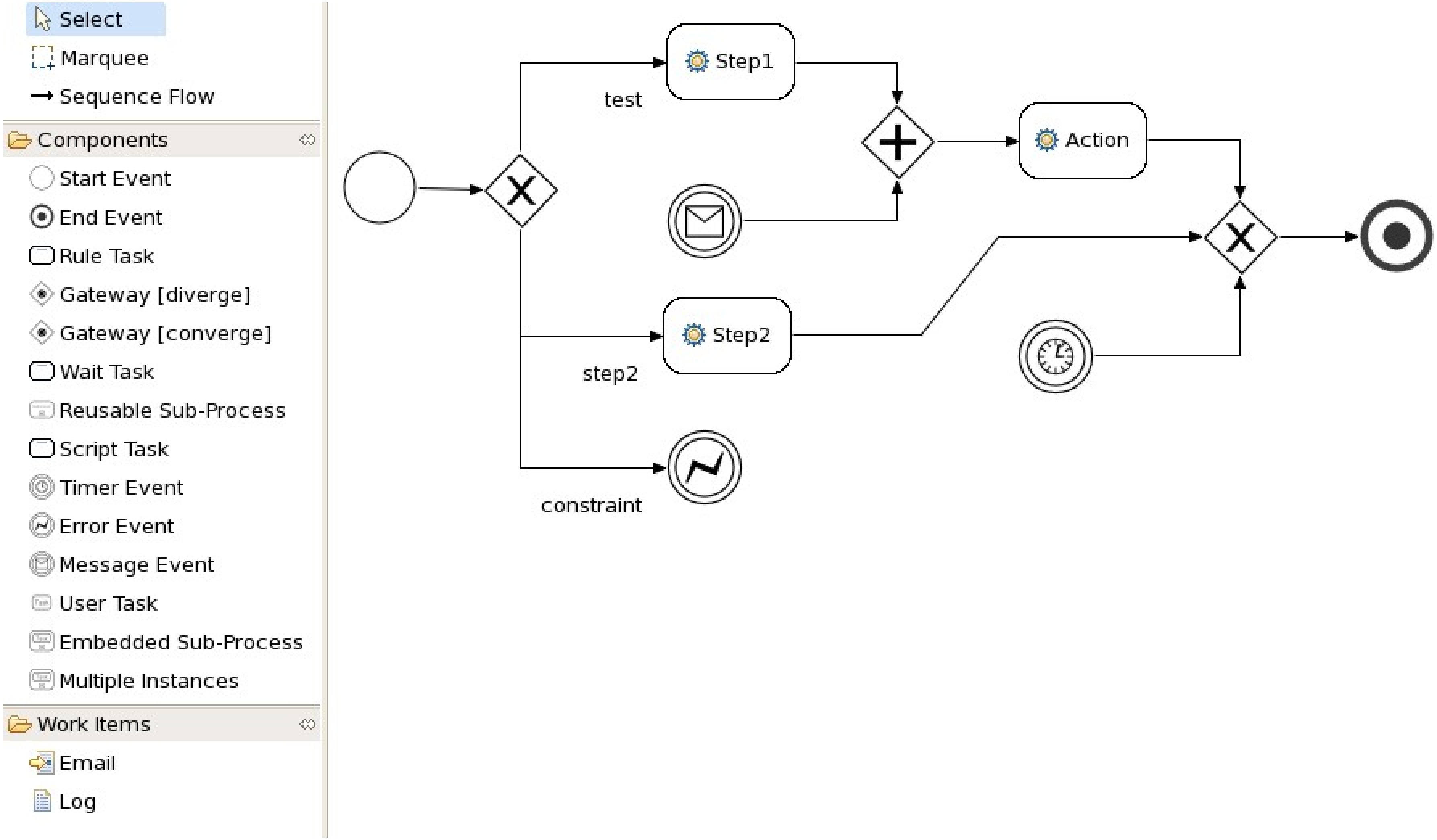
Drools Flow: integrated debug and audit



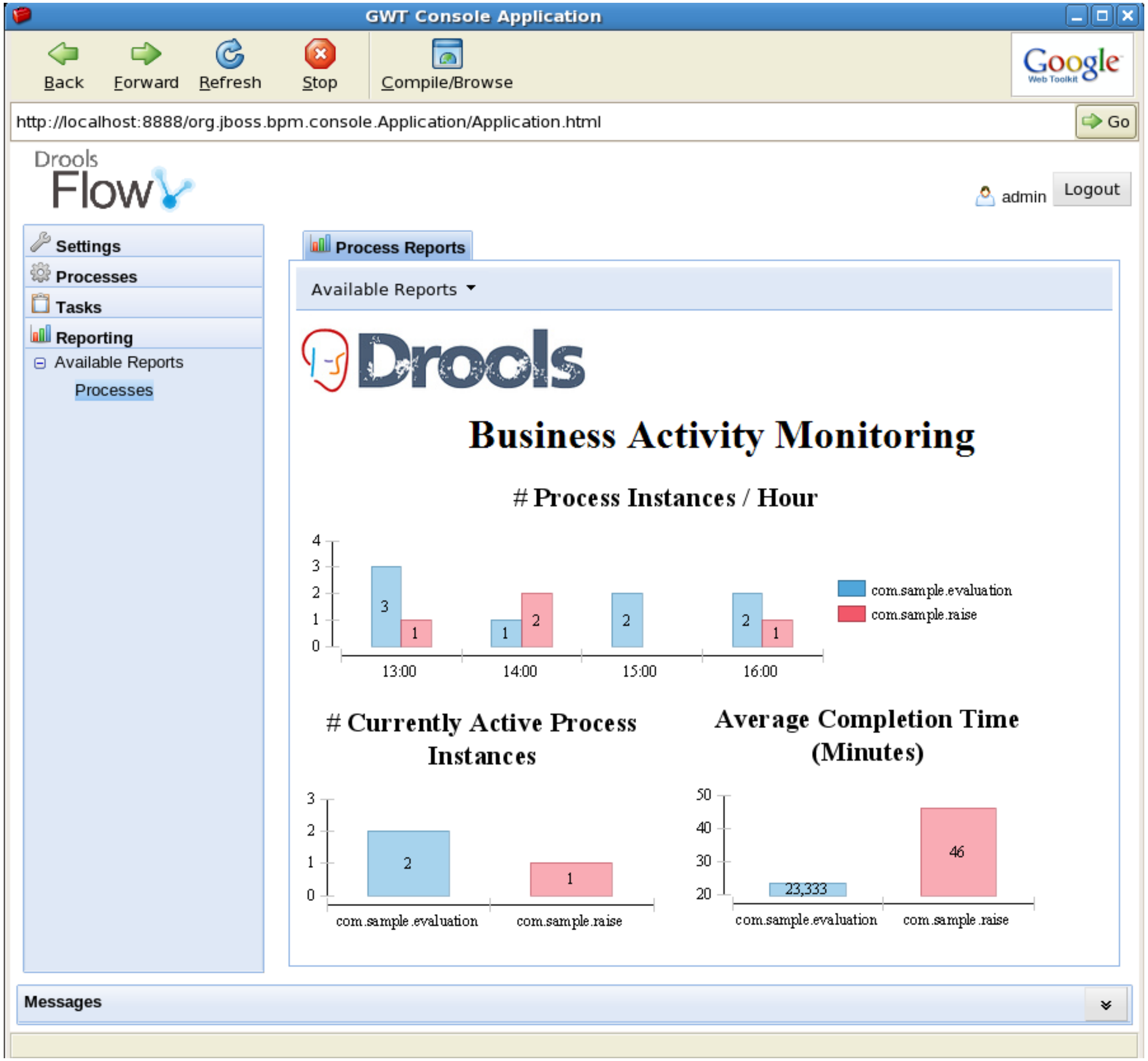


```
Object inserted (1): com.sample.RuleFlowTest$Message@17200b4
    Activation created: Rule Hello World m=com.sample.RuleFlowTest$Message@17200b4(1); message=Hello World(1)
RuleFlow started: ruleflow[com.sample.ruleflow]
    RuleFlow node triggered: Start in process ruleflow[com.sample.ruleflow]
        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]
```

Drools Flow: BPMN



Drools Flow: Business Activity Monitoring



Drools Business Logic Integration Platform

Complex Event Processing

“Complex Event Processing, or CEP, is primarily an event processing concept that deals with the task of processing multiple events with the goal of identifying the meaningful events within the event cloud.

CEP employs techniques such as detection of complex patterns of many events, event correlation and abstraction, event hierarchies, and relationships between events such as causality, membership, and timing, and event-driven processes.”

-- wikipedia

Complex Event Processing

- **A few characteristics of common CEP scenarios:**
  - Huge amount of events, but only a few of real interest
  - Usually events are immutable
  - Usually queries/rules have to run in reactive mode
  - Strong temporal relationships between events
  - Individual events are usually not important
  - The composition and aggregation of events is important

Drools Fusion: Characteristics

- Support processing high volume **Streams** of Events
- Typically fed from **SOA endpoints** such as JMS Queues, Web Service calls, Databases, flat files, or sockets.
- An **Event** is a record of state change. It is something that already happened, and the past cannot be changed, events are immutables.

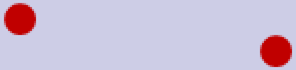
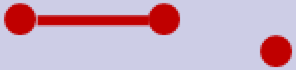
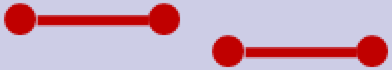
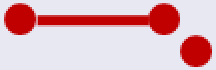
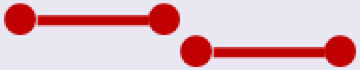
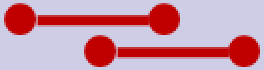
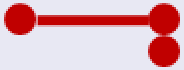
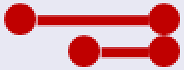
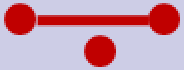
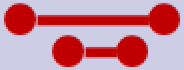
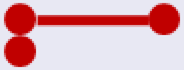
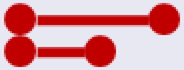
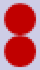
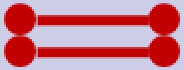
Drools Fusion: Enables...

- **Event Detection:**
  - From an event cloud or set of streams, select all the meaningful events, and only them.
- **[Temporal] Event Correlation:**
  - Ability to correlate events and facts declaring both temporal and non-temporal constraints between them.
  - Ability to reason over event aggregation.
- **Event Abstraction:**
  - Ability to compose complex events from atomic events AND reason over them

Drools Fusion: Features

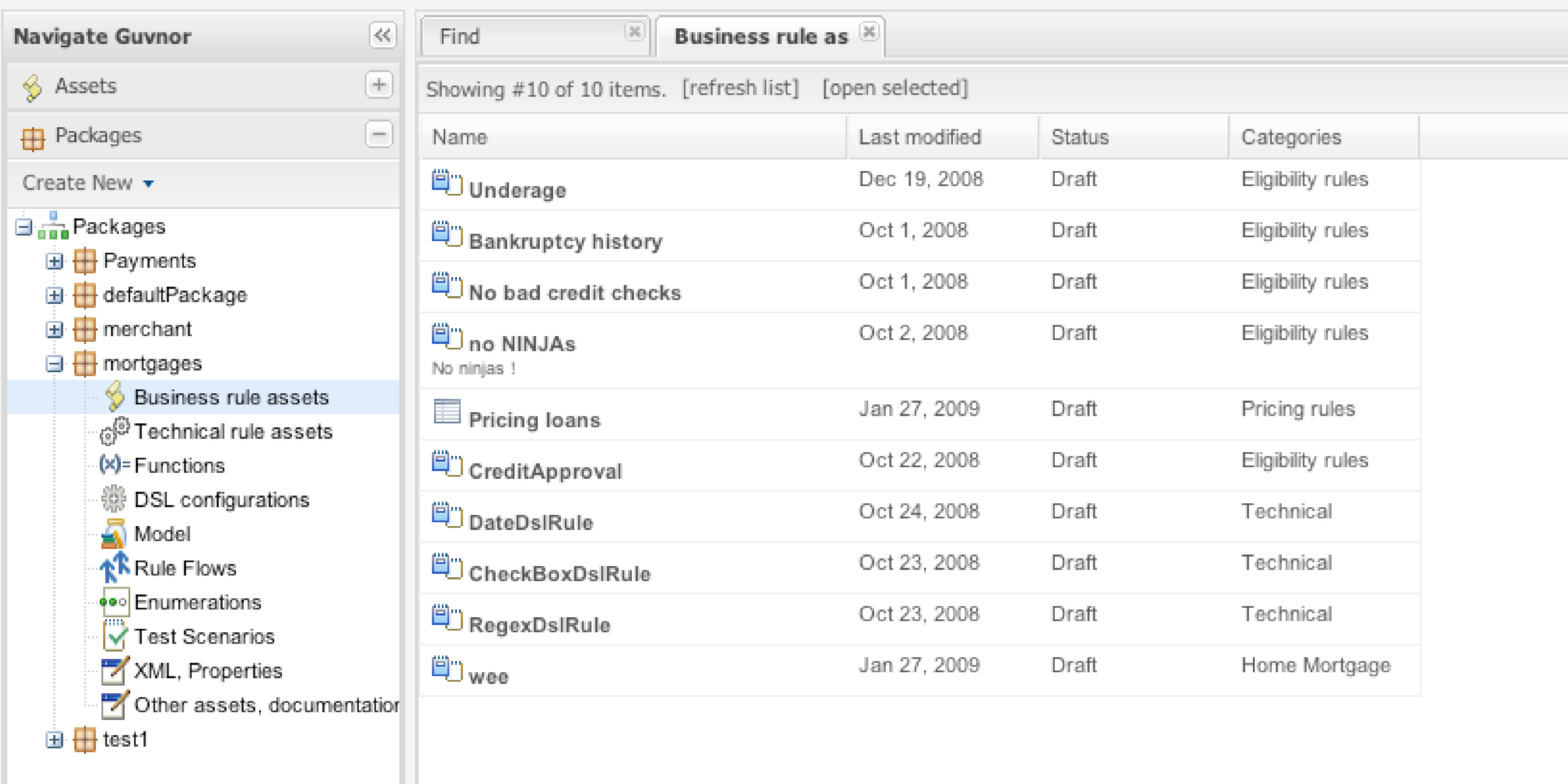
- **Engine Extension for Event Processing and Temporal Reasoning:**
  - New: Support to Event Semantics
  - New: Event Detection, Correlation and Composition
  - New: Support to both Cloud and Stream modes
  - New: Support to Stream Processing
  - New: Temporal Constraints
  - New: Sliding Windows
  - New: Session Clock
  - New: RuleBase Partitioning and Multithread: scalability
  - New: Active Queries and Rules
  - New: Data Loaders

Drools Fusion: Temporal Reasoning

	Point-Point	Point-Interval	Interval-Interval
A before B			
A meets B			
A overlaps B			
A finishes B			
A includes B			
A starts B			
A coincides B			

Drools Business Logic Integration Platform

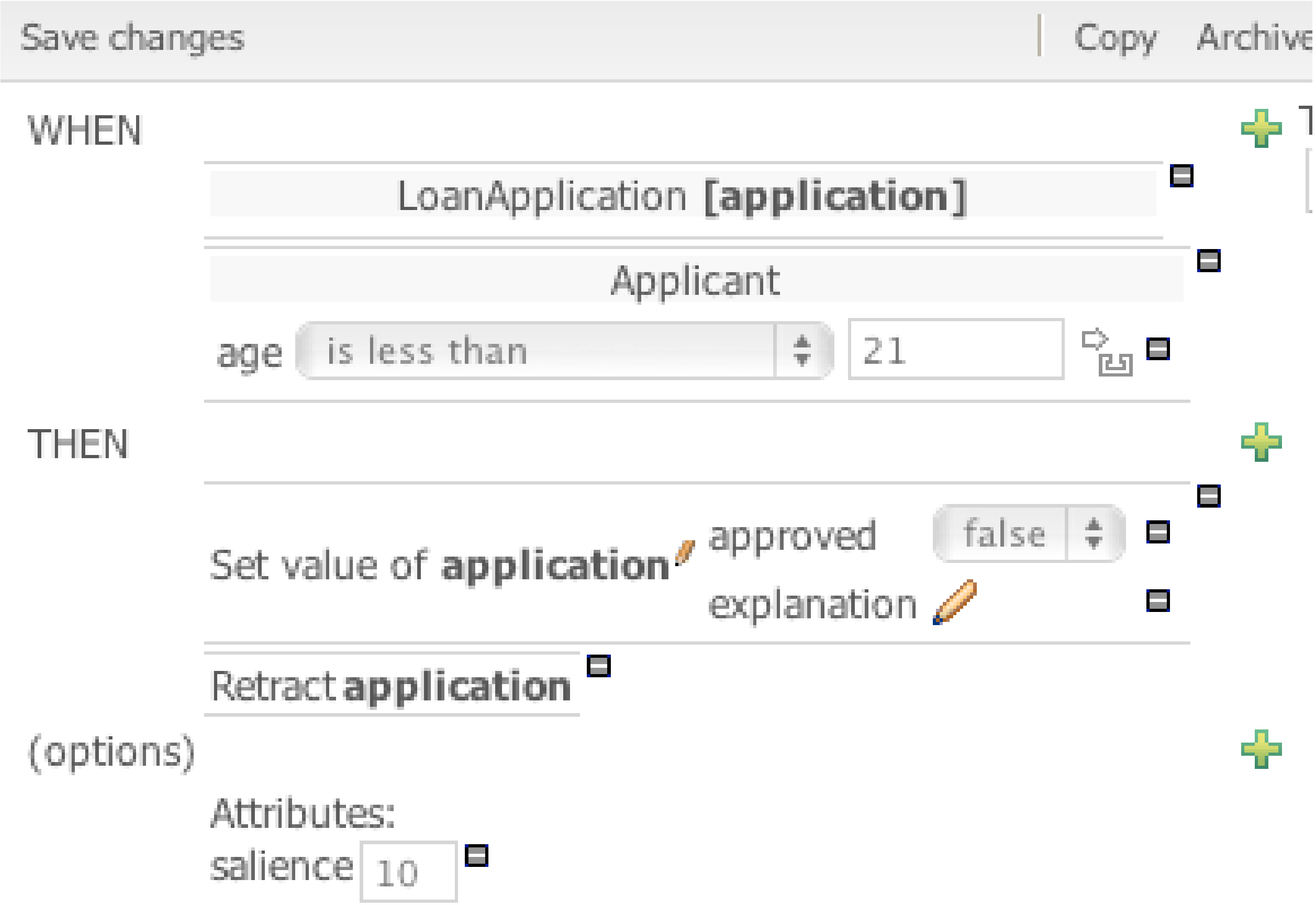
Drools Guvnor: Business Knowledge Governance



Showing #10 of 10 items.

Name	Last modified	Status	Categories
Underage	Dec 19, 2008	Draft	Eligibility rules
Bankruptcy history	Oct 1, 2008	Draft	Eligibility rules
No bad credit checks	Oct 1, 2008	Draft	Eligibility rules
no NINJAs No ninjas !	Oct 2, 2008	Draft	Eligibility rules
Pricing loans	Jan 27, 2009	Draft	Pricing rules
CreditApproval	Oct 22, 2008	Draft	Eligibility rules
DateDslRule	Oct 24, 2008	Draft	Technical
CheckBoxDslRule	Oct 23, 2008	Draft	Technical
RegexDslRule	Oct 23, 2008	Draft	Technical
wee	Jan 27, 2009	Draft	Home Mortgage

Drools Guvnor: Web authoring tools



WHEN

LoanApplication **[application]**

Applicant

age is less than 21

THEN

Set value of **application** approved false

explanation

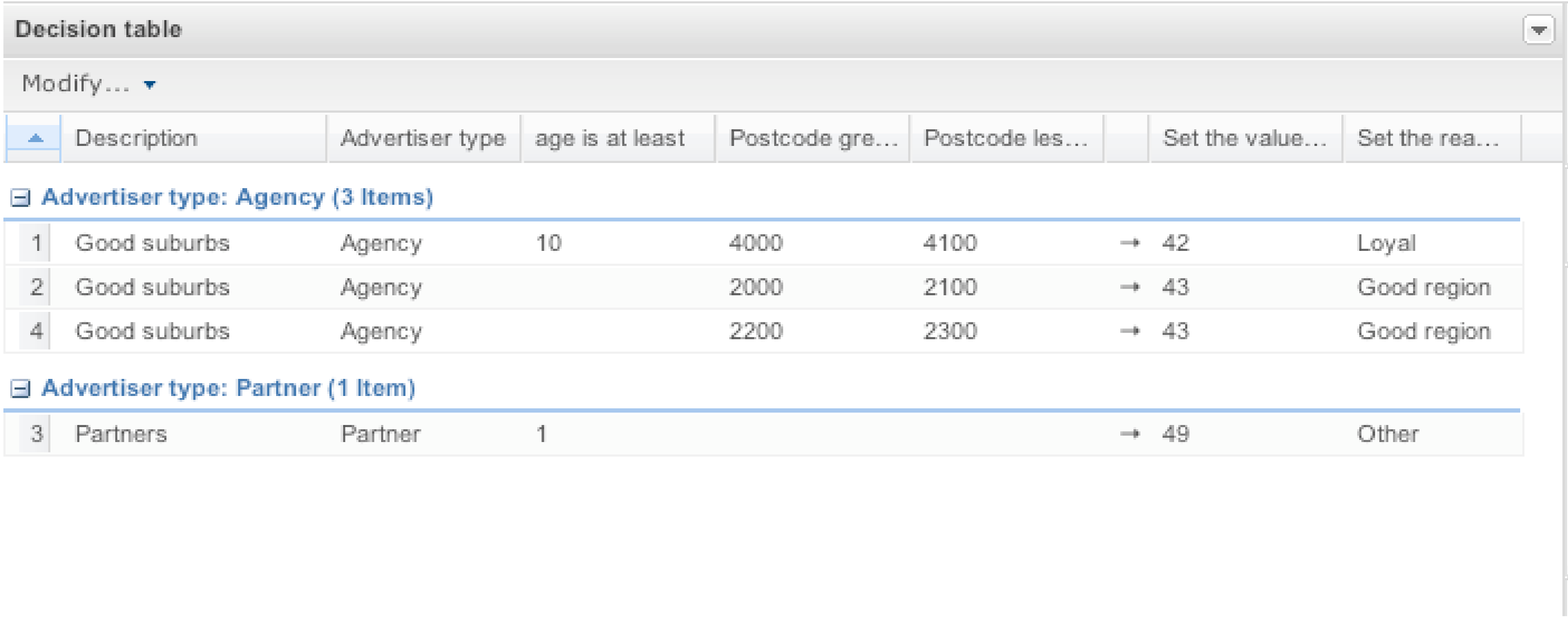
Retract **application**

(options)

Attributes:

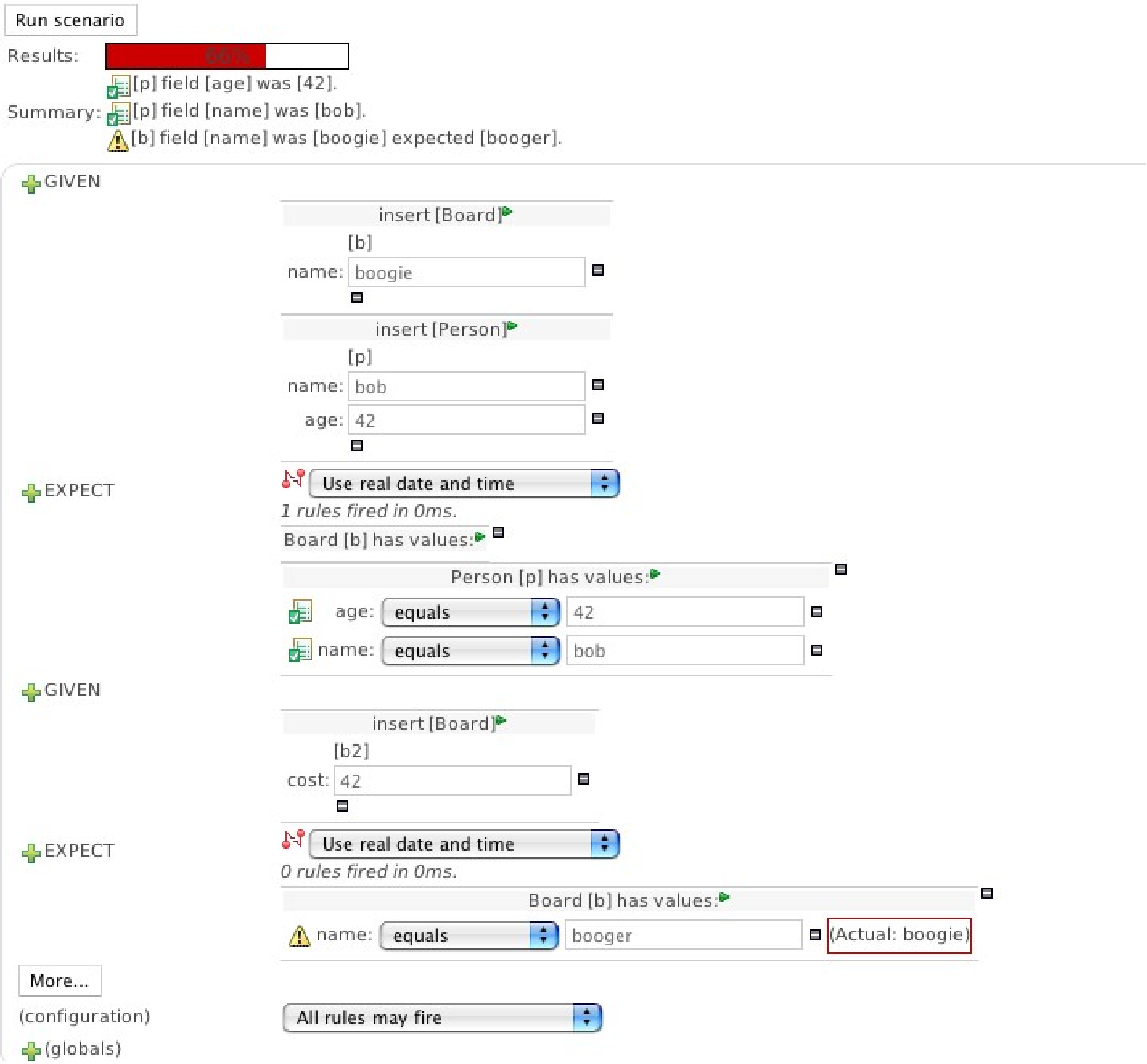
salience 10

Drools Guvnor: Web decision tables



	Description	Advertiser type	age is at least	Postcode gre...	Postcode les...		Set the value...	Set the rea...
⊟ Advertiser type: Agency (3 Items)								
1	Good suburbs	Agency	10	4000	4100	→	42	Loyal
2	Good suburbs	Agency		2000	2100	→	43	Good region
4	Good suburbs	Agency		2200	2300	→	43	Good region
⊟ Advertiser type: Partner (1 Item)								
3	Partners	Partner	1			→	49	Other

Drools Guvnor: Integrated Testing



Results: 66%

Summary:
✓ [p] field [age] was [42].
✓ [p] field [name] was [bob].
⚠ [b] field [name] was [boogie] expected [booger].

GIVEN

insert [Board]
[b]
name: boogie

insert [Person]
[p]
name: bob
age: 42

EXPECT

Use real date and time
1 rules fired in 0ms.
Board [b] has values:

Person [p] has values:
✓ age: equals 42
✓ name: equals bob

GIVEN

insert [Board]
[b2]
cost: 42

EXPECT

Use real date and time
0 rules fired in 0ms.

Board [b] has values:
⚠ name: equals booger (Actual: boogie)

(configuration) All rules may fire

(globals)

Drools: Demo

Drools
Business Logic integration Platform /
Guvnor
Expert
Fusion
Flow

QUESTIONS?

TELL US WHAT YOU THINK:
REDHAT.COM/JBOSSWORLD-SURVEY