



JBoss Drools - Viva Le Drools

Declarative Behavioural Modelling

An Integrated AI approach



Mark Proctor

Project Lead

- 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 Introduction
- Processes
 - RuleFlow
- Rules
 - Forward Chaining
 - Backwards Chaining
- Temporal Reasoning
 - CEP / ESP
- BAM
- Agents
- Uncertainty Systems to express truth degrees
- Solver

Debug - StateExampleUsingSalienc... - Eclipse SDK

File Edit Navigate Search Project Run Window Help

100%

Debug

StateExampleUsingSalienc... [Drools Application]

- org.drools.examples.StateExampleUsingSalienc... at localhost:4861
- Thread [main] (Suspended (breakpoint at line 8 in Rule_A_to_B_0))
 - Rule_A_to_B_0.consequence(KnowledgeHelper, State, FactHandle) line: 21
 - Rule_A_to_B_0ConsequenceInvoker.evaluate(KnowledgeHelper, WorkingMemory) line: 22
 - DefaultAgenda.fireActivation(Activation) line: not available
 - DefaultAgenda.fireNextItem(AgendaFilter) line: not available
 - ReteooWorkingMemory(AbstractWorkingMemory).fireAllRules(AgendaFilter) line: not available
 - ReteooWorkingMemory(AbstractWorkingMemory).fireAllRules() line: not available
 - StateExampleUsingSalienc... main(String[]) line: 47

Variables

Name	Value
b	State (id=1268)
changes	PropertyChangeSupport (id=1297)
name	"B"
state	1

StateExampleUsingSalienc... drl

```

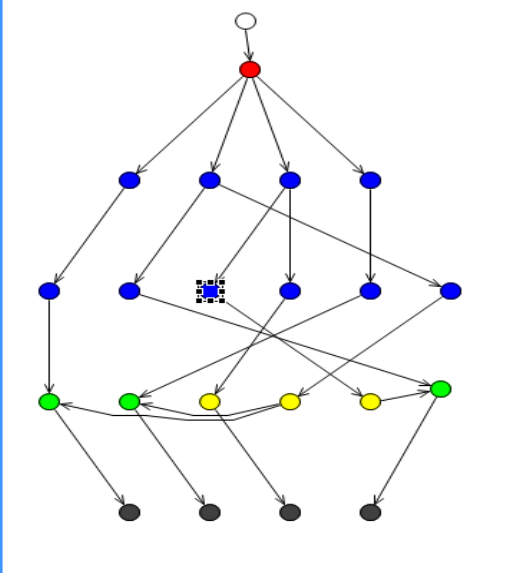
import org.drools.examples.State;

rule Bootstrap
when
  a : State(name == "A", state == State.NOTRUN )
then
  System.out.println(a.getName() + " finished" );
  a.setState( State.FINISHED );
end

rule "A to B"
when
  State(name == "A", state == State.FINISHED )
  b : State(name == "B", state == State.NOTRUN )
then
  b.setState( State.FINISHED );
  System.out.println(b.getName() + " finished" );
end

rule "B to C"
salience 10
when
  State(name == "B", state == State.FINISHED )
  c : State(name == "C", state == State.NOTRUN )
then
  System.out.println(c.getName() + " finished" );
end
    
```

StateExampleUsingSalienc... drl



Properties

Property	Value
Constraint	[LiteralConstraint fieldExtr...
Evaluator	Integer ==
Field Name	state
Name	Alpha BaseVertex
Value	1

Outline

- org.drools.examples
 - A to B
 - B to C
 - B to D
 - Bootstrap
 - org.drools.examples.State

Text Editor | Rete Tree

Text Editor | Rete Tree

Global Data View

The selected working memory has no globals defined.

A finished

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]

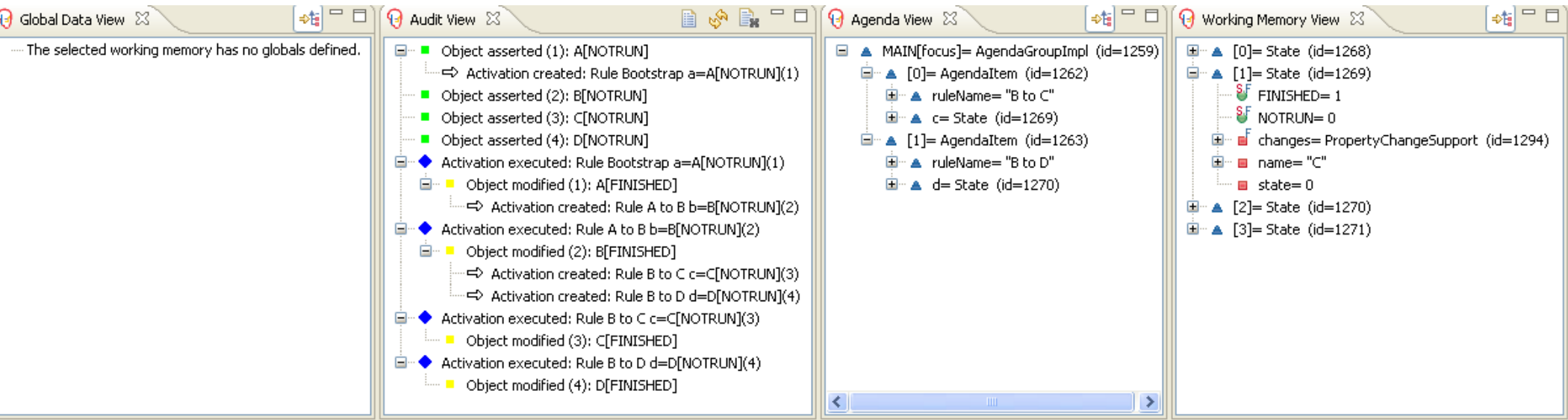
Agenda View

- 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)

Working Memory View

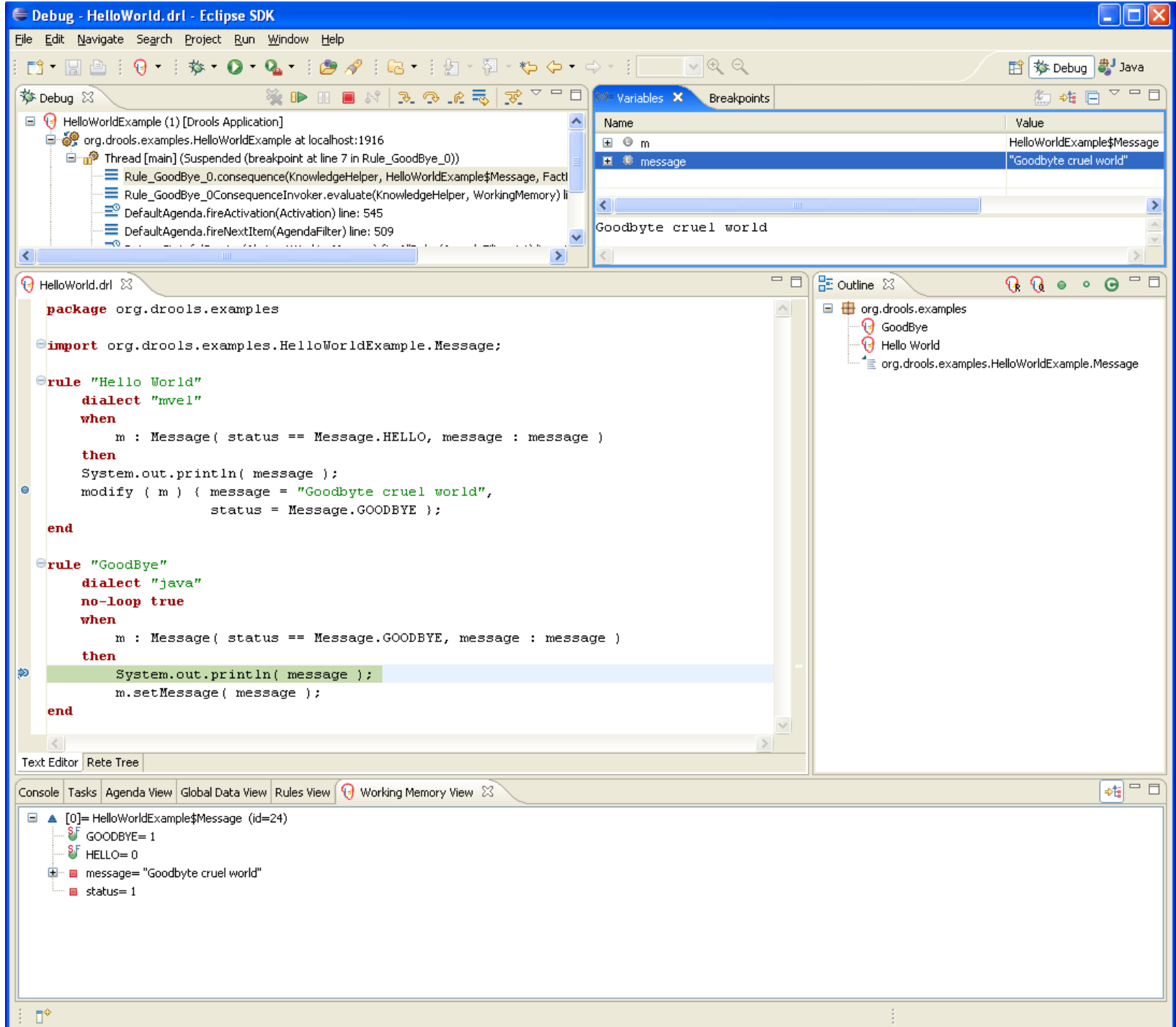
- [0]= State (id=1268)
- [1]= State (id=1269)
 - FINISHED= 1
 - NOTRUN= 0
 - changes= PropertyChangeSupport (id=1294)
 - name= "C"
 - state= 0
- [2]= State (id=1270)
- [3]= State (id=1271)

Writable Insert 21 : 1



The screenshot displays the Eclipse IDE interface with four views open:

- Global Data View:** Shows the message "The selected working memory has no globals defined."
- Audit View:** Displays a sequence of events:
 - 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:** Shows the current state of the agenda:
 - 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)
- Working Memory View:** Shows the current state of the working memory:
 - [0]= State (id=1268)
 - [1]= State (id=1269)
 - FINISHED= 1
 - NOTRUN= 0
 - changes= PropertyChangeSupport (id=1294)
 - name= "C"
 - state= 0
 - [2]= State (id=1270)
 - [3]= State (id=1271)



The screenshot shows the Eclipse IDE interface with the following components:

- Debugger:** Shows the execution stack for `org.drools.examples.HelloWorldExample` at `localhost:1916`. The `main` thread is suspended at a breakpoint in `Rule_GoodBye_0`. The stack trace includes `Rule_GoodBye_0.consequence`, `Rule_GoodBye_0ConsequenceInvoker.evaluate`, `DefaultAgenda.fireActivation` (line 545), and `DefaultAgenda.fireNextItem` (line 509).
- Variables View:** Displays the current state of variables:

Name	Value
m	HelloWorldExample\$Message
message	"Goodbye cruel world"
- Code Editor:** Shows the `HelloWorld.drl` file with two rules:

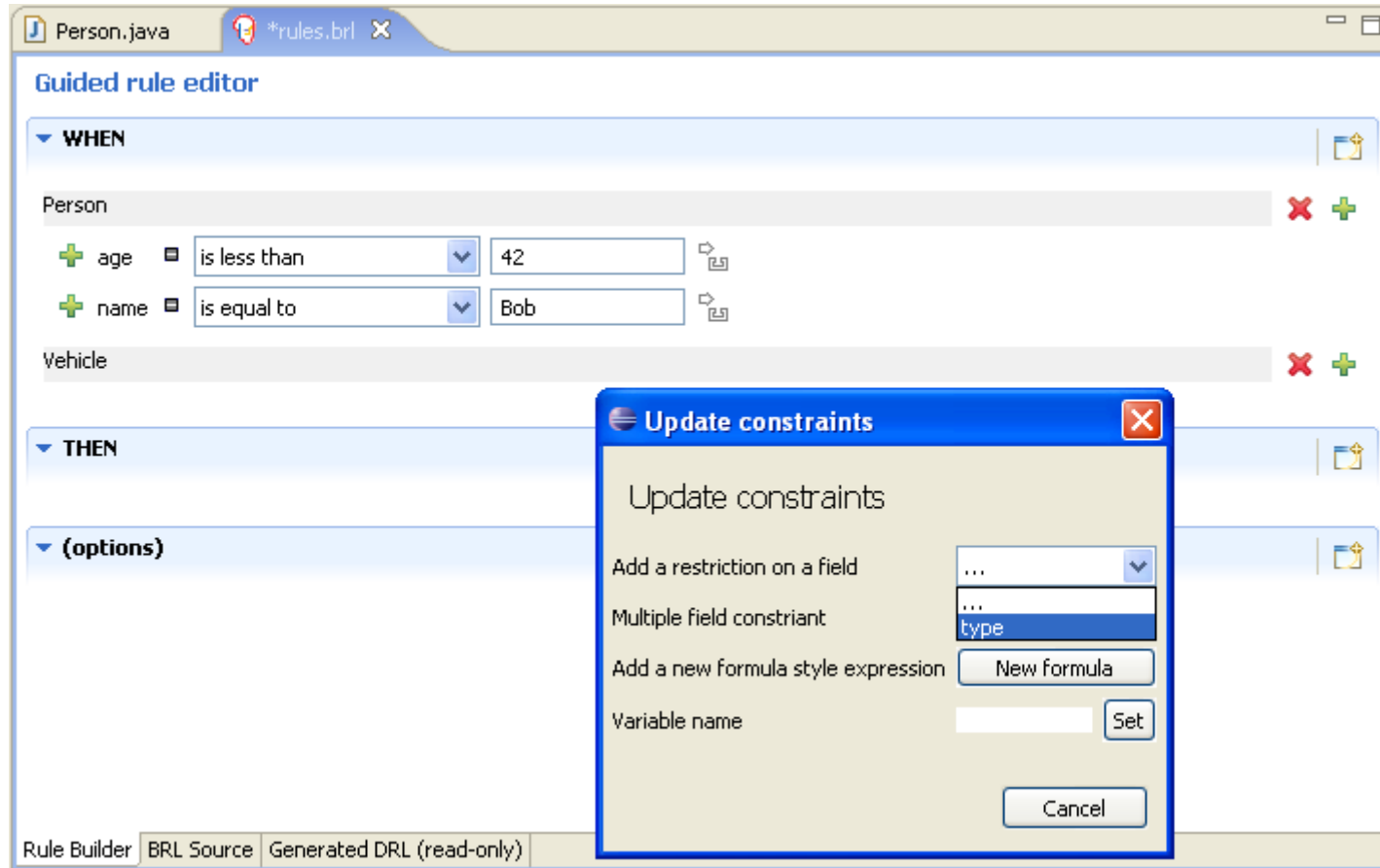
```
package org.drools.examples

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 = "Goodbye 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
```
- Outline View:** Shows the project structure:
 - org.drools.examples
 - GoodBye
 - Hello World
 - org.drools.examples.HelloWorldExample.Message
- Console:** Shows the output of the application:

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

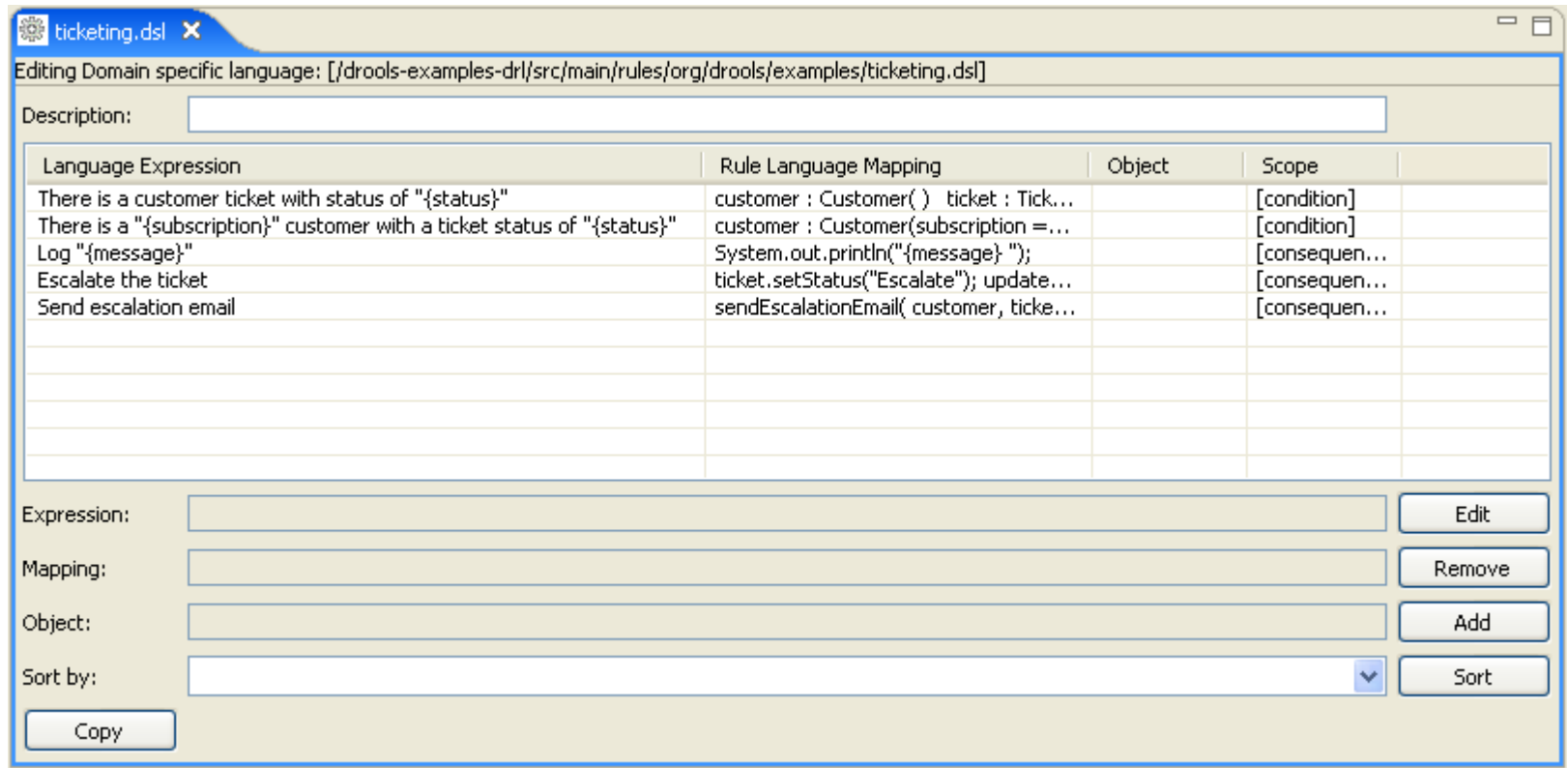


The screenshot displays the Eclipse IDE's Guided Rule Editor. The main window is titled "Guided rule editor" and shows a rule configuration for "Person" and "Vehicle". The "WHEN" section is expanded, showing two constraints: "age is less than 42" and "name is equal to Bob". The "THEN" section is collapsed, and the "(options)" section is also collapsed. A dialog box titled "Update constraints" is open in the foreground, allowing the user to modify the constraints. The dialog box has a title bar with a close button and contains the following fields and buttons:

- Update constraints
- Add a restriction on a field: ...
- Multiple field constraint: ... type
- Add a new formula style expression: New formula
- Variable name: Set
- Cancel

The bottom of the window shows the "Rule Builder" tab, with "BRL Source" and "Generated DRL (read-only)" sub-tabs.

```
rule "Driver in unsafe area for marginal age"  
  when  
    Policy type is 'COMPREHENSIVE'  
    Driver is less than 25 years old  
    Driver has a location risk profile of 'HIGH'  
  
  then  
    <> Driver has a location risk profile of '{risk}'  
    <> Driver has an age of at least {age}  
    <> Driver has had more than {prior} prior claims  
    <> Driver has had {number} prior claims  
  end  
  
rule "Driver unsafe for marginal age driver in high risk area"  
  when  
    <> Driver is between {lower} and {upper} years old  
    <> Driver is greater than {age} years old  
    <> Driver is less than {age} years old  
    <> Policy has not been rejected  
    <> Policy type is '{type}'  
  
  then  
    Reject Policy with explanation : 'Driver in that area is too risky -'  
  end  
  
rule "Driver unsafe for third party"  
  when  
    Policy type is 'THIRD_PARTY'  
    Driver has had more than 2 prior claims  
  end
```



The screenshot shows the Eclipse IDE interface for editing a Drools DSL file named 'ticketing.dsl'. The window title is 'ticketing.dsl' and the address bar shows the file path: [/drools-examples-drl/src/main/rules/org/drools/examples/ticketing.dsl].

Below the address bar is a 'Description:' field. The main area contains a table with the following columns: Language Expression, Rule Language Mapping, Object, and Scope.

Language Expression	Rule Language Mapping	Object	Scope
There is a customer ticket with status of "{status}"	customer : Customer() ticket : Tick...		[condition]
There is a "{subscription}" customer with a ticket status of "{status}"	customer : Customer(subscription =...		[condition]
Log "{message}"	System.out.println("{message} ");		[consequen...]
Escalate the ticket	ticket.setStatus("Escalate"); update...		[consequen...]
Send escalation email	sendEscalationEmail(customer, ticke...		[consequen...]

Below the table are several input fields and buttons:

- Expression: [Edit]
- Mapping: [Remove]
- Object: [Add]
- Sort by: [Sort]
- [Copy]



Decision Tables (Excel/OpenOffice)

	B	C	D	E	F	G	H
1							
4							
9	Base pricing rules	Age Bracket	Location risk profile	Number of prior claims	Policy type applying for	Base \$ AUD	Record Reason
10	Young safe package	18, 24	LOW	1	COMPREHENSIVE	450	
11			MED		FIRE_THEFT	200	Priors not relevant
12			MED	0	COMPREHENSIVE	300	
13			LOW		FIRE_THEFT	150	
14			LOW	0	COMPREHENSIVE	150	Safe driver discount
15	Young risk	18,24	MED	1	COMPREHENSIVE	700	
16		18,24	HIGH	0	COMPREHENSIVE	700	Location risk
17		18,24	HIGH		FIRE_THEFT	550	Location risk
18	Mature drivers	25,30		0	COMPREHENSIVE	120	Cheapest possible
19		25,30		1	COMPREHENSIVE	300	
20		25,30		2	COMPREHENSIVE	590	
21		25,35		3	THIRD PARTY	800	High risk

JBoss Business Rules Management System - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://localhost:8080/drools-jbrms/org.drools.brms.JBRMS/JBRMS.jsp#Rules

[Gmail](#)
[post to del.icio.us](#)
[my del.icio.us](#)
[Browse Project - JBo...](#)
[Google Calendar](#)
[Red Hat -- Intranet H...](#)
[Nopaste](#)
[Brisbane Times - New...](#)

JBoss Rules

Find and edit rules.

Rules: [Explore](#) [Rule_2](#) [MyDT](#) [Rule_1](#)

Info Packages Deployment Admin

Status: **[Draft]** [Save changes](#) [Copy](#) [Archive](#)

IF

- Person
 - age less than or equal to 42
 - age greater than 21
- Board [b]
- There is no Board
 - cost greater than 1200

THEN

- Set [b] cost 1200

(options)

[View source](#) [Validate](#)

<documentation>

Rule_1

Categories: Finance HR/Awards/QAS

Modified on: Thu 10 May 2007 03:16:47 PM EST
 by: michael
 Note: whoops
 Version: 5
 Created on: Thu 10 May 2007 10:49:41 AM EST
 Created by: michael
 Format: brxml

Package: DemoPackage

Subject:

Type:

External link:

Source:

Version history [View](#)

Done

JBoss Business Rules Management System

http://localhost:8080/drools-jbrms/org.drools.brms.JBRMS/JBRMS.jsp

User: fmeyer [Sign Out]

JBoss Rules

Administer the repository

Manage categories | Manage status

Create a new top level category.

Category name:

Description:

OK | Cancel

Categories aid in managing large numbers of rules/assets. A shallow hierarchy is recommended.

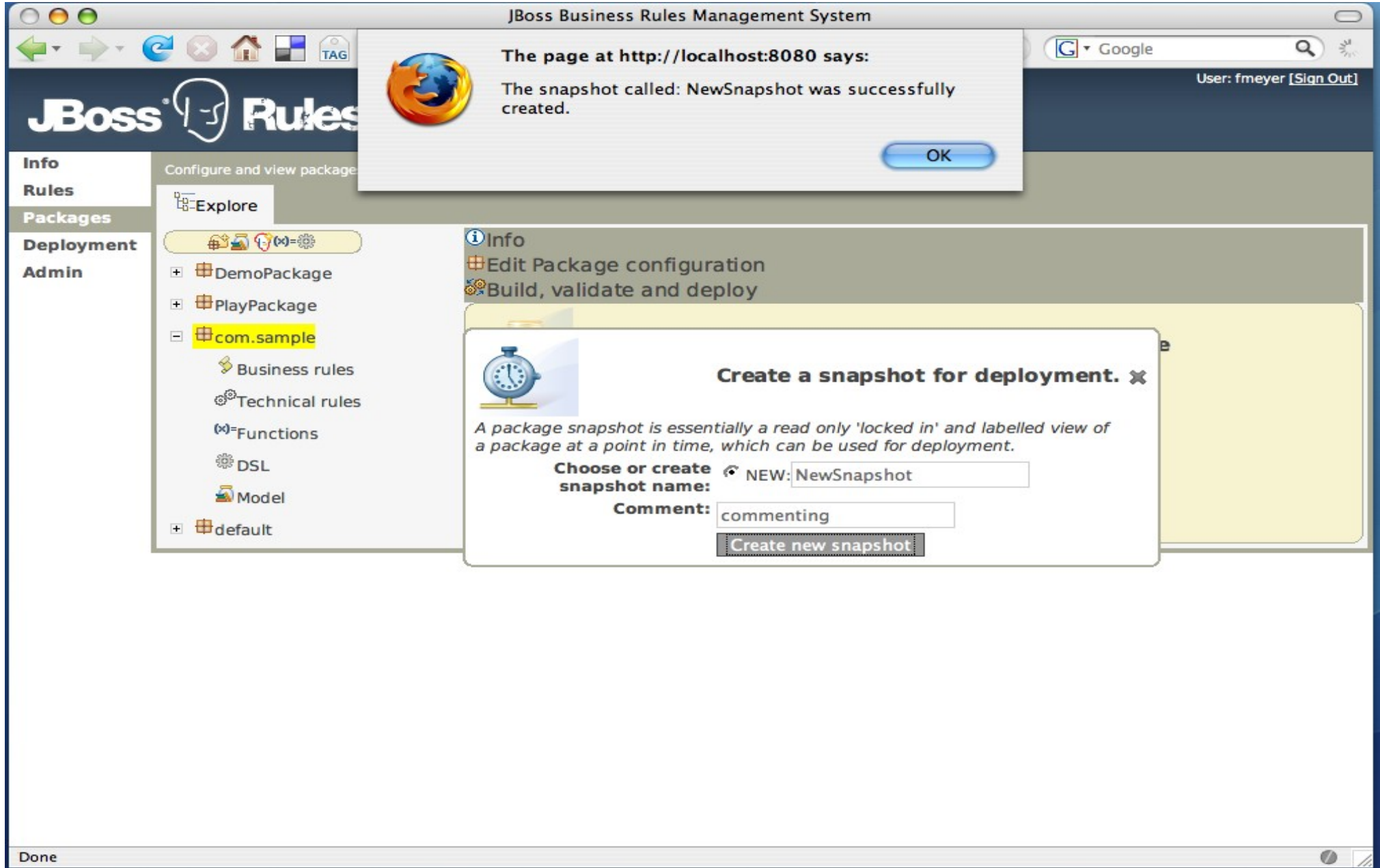
Current categories:

- + HR
- + Finance
- + Draft

Refresh view:

Create a new category:

Delete the currently selected category:



The screenshot shows the JBoss Business Rules Management System (BRMS) web interface. At the top, the title bar reads "JBoss Business Rules Management System". A notification window from the browser states: "The page at http://localhost:8080 says: The snapshot called: NewSnapshot was successfully created." with an "OK" button.

The main interface features a left-hand navigation menu with categories: Info, Rules, Packages, Deployment, and Admin. Under "Packages", the "com.sample" package is selected, showing sub-items: Business rules, Technical rules, Functions, DSL, and Model. The "default" package is also visible.

The main content area displays "Info" for the selected package, with options to "Edit Package configuration" and "Build, validate and deploy". A prominent yellow box titled "Create a snapshot for deployment." contains a clock icon and the following text: "A package snapshot is essentially a read only 'locked in' and labelled view of a package at a point in time, which can be used for deployment." Below this, there are input fields for "Choose or create snapshot name:" (with a radio button selected for "NEW:" and the value "NewSnapshot") and "Comment:" (with the value "commenting"). A "Create new snapshot" button is located at the bottom of this box.

The top right of the interface shows a search bar with "Google" and a user profile for "User: fmeyer [Sign Out]". The bottom left corner of the browser window shows "Done".

JBoss Business Rules Management System - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://localhost:8080/drools-jbrms/org.drools.brms.JBRMS/JBRMS.jsp#Rules

Gmail post to del.icio.us my del.icio.us Browse Project - JBo... Google Calendar Red Hat -- Intranet H... Nopaste Brisbane Times - New...

Viewing source for: Rule_1

```
rule "Rule_1"
  when
    Person( age <= 42 , age > 21 )
    b : Board( )
    not Board( cost > 1200 )
  then
    b.setCost( 1200 );
  end
```

Rule_1

Categories: Finance HR/Awards/QAS

Modified on: Thu 10 May 2007 03:16:47 PM EST
by: michael
Note: whoops
Version: 5
Created on: Thu 10 May 2007 10:49:41 AM EST
Created by: michael
Format: brxml

Package: DemoPackage

Subject:

Type:

External link:

Source:

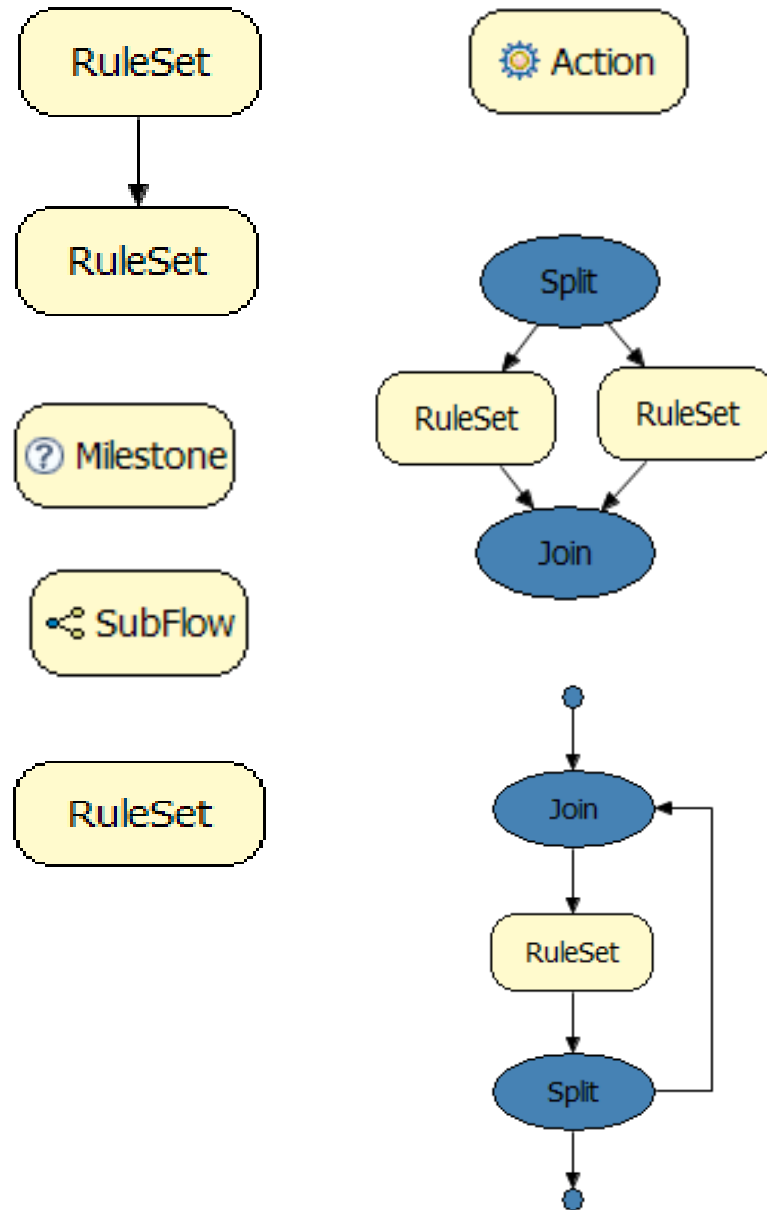
Version history

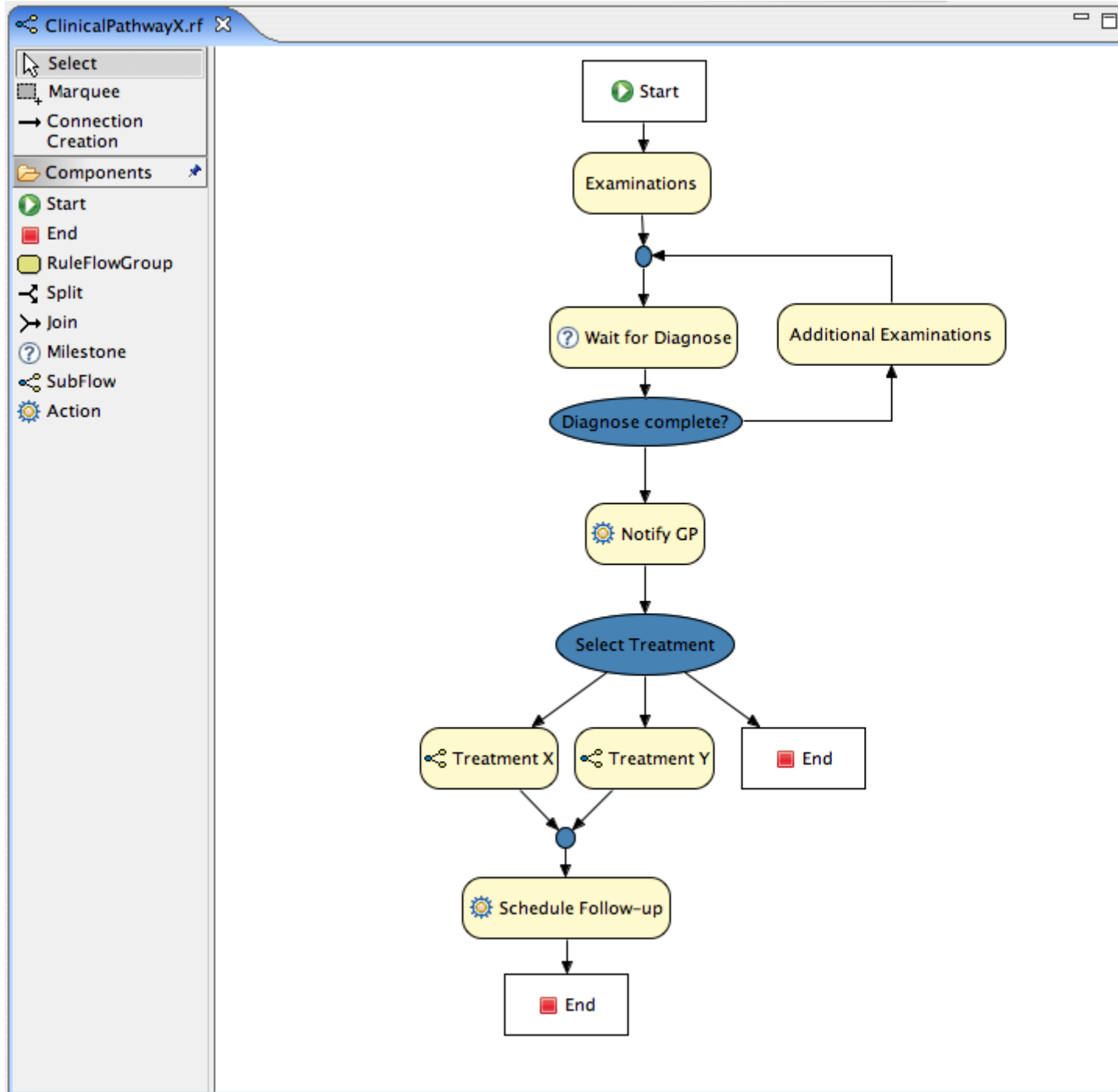
View source Validate

Done

- Unifies Rules and Processes in a single engine
 - Rules (LHS When) and expressions can be used in splits, milestones etc
 - creates a much richer model
 - Rules and Processes see, reason and react and process the same data
 - Do not have send messages between two different engines
 - Multiple instances, of different processes, can be executing at the same time in a single engine.
 - Processes and Rules interactive with each other.
 - A Process or Rule can change data, which can impact how another rule or process is executing.
 - Integrated Tooling and APIs
 - Single api for execution
 - Audit logging and visual Audit tools
 - Single server side tooling for storage, configuration and management and deployment

- Rule set nodes
- Control flow
 - Sequence
 - Parallelism (split / join)
 - Choice
- Nodes
 - Actions
 - Milestone (= state)
 - Subflows
 - Looping





Java - NumberGuess.drl - Eclipse SDK - /Users/fmeyer/projects/droolsprojects

***NumberGuess.rf**

```

graph TD
    Start([Start]) --> Join1((More guesses Join))
    Join1 --> Guess[Guess]
    Guess --> Correct{Guess correct?}
    Correct --> CorrectAction[Guess Correct]
    CorrectAction --> Join2((No more guesses Join))
    Join2 --> End([End])
    Correct --> TooHigh[Too High]
    Correct --> TooLow[Too Low]
    TooHigh --> Incorr{Incorrect guess}
    TooLow --> Incorr
    Incorr --> IncorrAction[Guess incorr...]
    IncorrAction --> MoreGuesses{More Guesses?}
    MoreGuesses --> Join1
    
```

NumberGuess.drl

```

26     insert( new Guess( i ) );
27 end
28
29 rule "Record the highest Guess"
30     ruleflow-group "Too High"
31     no-loop
32     when
33         game : Game( biggestGuess : biggest )
34         Guess( $value : value > biggestGuess )
35     then
36         modify ( game ) { biggest = $value };
37     end
38
39 rule "Record the lowest Guess"
40     ruleflow-group "Too Low"
41     no-loop
42     when
43         Game( smallestGuess : smallest )
44         Guess( $value : value < smallestGuess )
45     then
46         modify ( game ) { smallest = $value };
47     end
48
49 rule "Guess incorrect, retract Guess"
50     ruleflow-group "Guess incorrect"
51     when
52         guess : Guess()
53     then
54         retract( guess );
55     end
56
57
58 rule "No more Guesses notification"
    
```

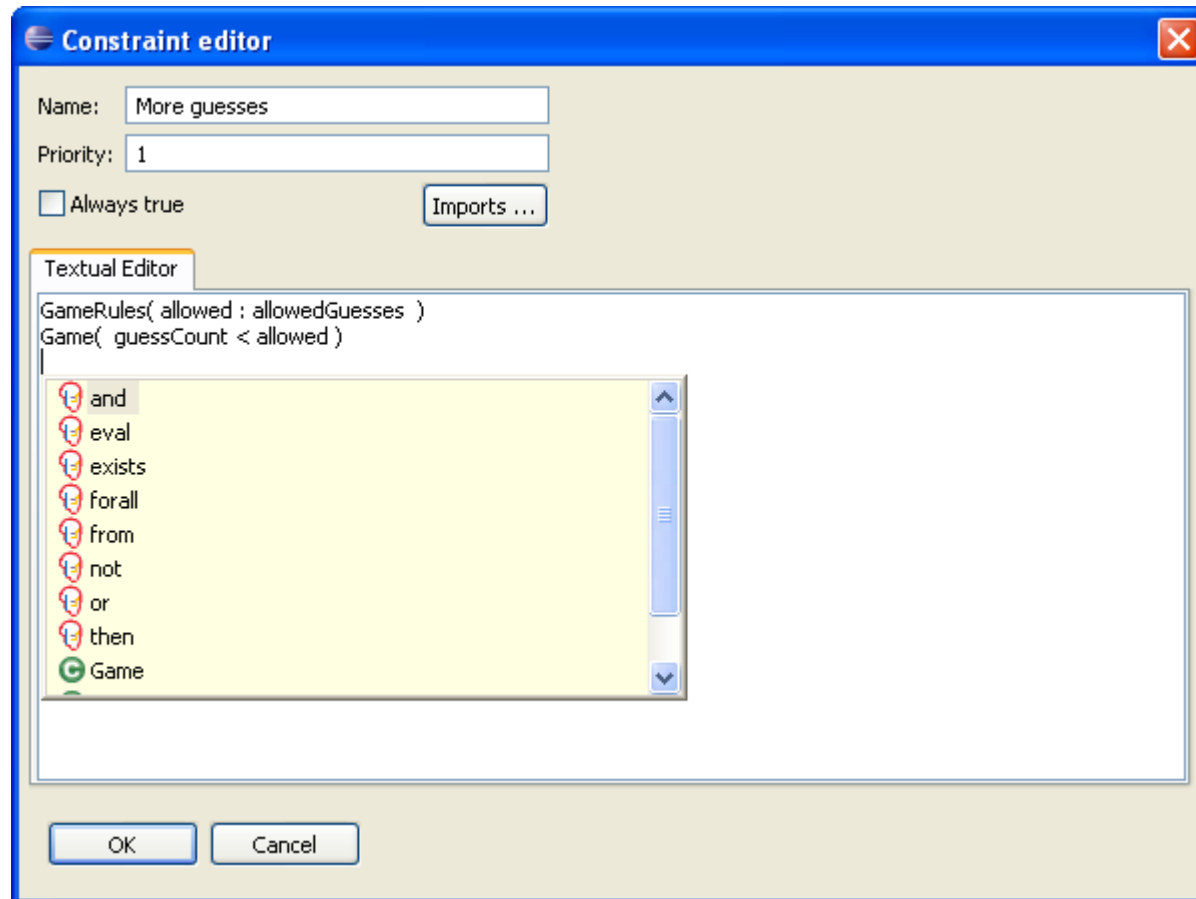
Package Explorer

- org.acme.insurance
- org.benchmarks.waltz
- org.drools
- org.drools.benchmark.manners
- org.drools.benchmark.waltzdb
- org.drools.compiler
- org.drools.examples
 - A to B
 - A to B
 - Apply 10% discount if total purchasess is over 100
 - B to C

Outline


- org.drools.examples
 - Get user Guess
 - Guess incorrect, retract Guess
 - No more Guesses notification
 - Record the highest Guess
 - Record the lowest Guess
 - java.io.BufferedReader
 - java.io.InputStreamReader
 - org.drools.examples.NumberGuessExample.Game
 - org.drools.examples.NumberGuessExample.GameRules
 - org.drools.examples.NumberGuessExample.Guess

Writable | Insert | 60 : 9



- ◆ Activation executed: Rule Start Clinical Pathway X if diagnosed d=Diagnose: Diagnose disease X: Type unknown(2)
 - Object removed (2): Diagnose: Diagnose disease X: Type unknown
 - ↳ Activation cancelled: Rule RuleFlow-org.drools.examples.cdss.ClinicalPathwayX-16-17
 - ↳ Activation cancelled: Rule Remove old diagnose d=Diagnose: Diagnose disease X: Type unknown(2)
 - ↳ Activation cancelled: Rule RuleFlow-org.drools.examples.cdss.ClinicalPathwayX-12
 - 🔗 RuleFlowGroup activated: Examinations[size=2]
 - 🔗 RuleFlow started: ClinicalPathwayX[org.drools.examples.cdss.ClinicalPathwayX]
- ◆ Activation executed: Rule Examination1
- ◆ Activation executed: Rule Examination2
- 🔗 RuleFlowGroup deactivated: Examinations[size=0]
- 🔗 RuleFlowGroup activated: AdditionalExaminations[size=2]
- Object inserted (2): Diagnose: Diagnose disease X: Type unknown
 - ⇒ Activation created: Rule Start Clinical Pathway X if diagnosed d=Diagnose: Diagnose disease X: Type unknown(2)
 - ⇒ Activation created: Rule RuleFlow-org.drools.examples.cdss.ClinicalPathwayX-16-17
 - ⇒ Activation created: Rule Remove old diagnose d=Diagnose: Diagnose disease X: Type unknown(2)
 - ⇒ Activation created: Rule RuleFlow-org.drools.examples.cdss.ClinicalPathwayX-12
- ◆ Activation executed: Rule Remove old diagnose d=Diagnose: Diagnose disease X: Type unknown(2)
 - Object removed (2): Diagnose: Diagnose disease X: Type unknown
 - ↳ Activation cancelled: Rule Start Clinical Pathway X if diagnosed d=Diagnose: Diagnose disease X: Type unknown(2)
 - ↳ Activation cancelled: Rule RuleFlow-org.drools.examples.cdss.ClinicalPathwayX-16-17
 - ↳ Activation cancelled: Rule RuleFlow-org.drools.examples.cdss.ClinicalPathwayX-12
 - ◆ Activation executed: Rule Examination3
 - 🔗 RuleFlowGroup deactivated: AdditionalExaminations[size=0]
 - 🔗 RuleFlow completed: TreatmentY[org.drools.examples.cdss.TreatmentY]
 - 🔗 RuleFlow started: TreatmentY[org.drools.examples.cdss.TreatmentY]
 - 🔗 RuleFlow completed: ClinicalPathwayX[org.drools.examples.cdss.ClinicalPathwayX]
- Object inserted (2): Diagnose: Diagnose disease X: Type 2

Explore



- org.acme.insurance.base
 - Business rule assets
 - Technical rule assets
 - Functions
 - DSL
 - Model

15 items.

	Name	Last modified	Status
⚙️	Insurance extra itens percent	Sep 20, 2007	Production
⚙️	Insurance Calcule	Sep 20, 2007	Production
⚙️	Driver is underage	Sep 20, 2007	Production
⚙️	New licenced Driver	Sep 20, 2007	Production
⚙️	Driver Single Young Male Driver factor	Aug 28, 2007	Production
⚙️	Driver Mature Married With Young Child factor	Aug 28, 2007	Production
⚙️	Priory Claimed Driver	Aug 28, 2007	Production
⚙️	Day Vehicle Place	Aug 28, 2007	Production
⚙️	Night Vehicle Place	Aug 28, 2007	Production
⚙️	Driver wants an extra Car	Aug 28, 2007	Production
⚙️	Driver wants glass coverage	Aug 28, 2007	Production
⚙️	Driver wants non related expenses coverage	Aug 28, 2007	Production
👤	insuranceProcess	Aug 28, 2007	Production
⚙️	approve	Aug 28, 2007	Production
⚙️	rejection	Aug 28, 2007	Production

- Current only supports forward chaining
 - Plans to support both backward/forward chaining
- when
 - `$p : Person($age : age == 17) // from Working Memory`
 - `legalDriver($age, $state:) // executes correlated query`
 - `// and unifies results with`
 - `// current tuple, queries`
 - `// can nested and recursive`
- then
 - `// prints all valid states`
 - `System.out.println($p.name + ": " + $state);`

- Reasoning over absolute time windows
 - This is the common case of reasoning over a slide time-window and/or aggregation of values: "when the average transaction throughput calculated over 1 minute goes below the threshold, raise the alarm"

rule "absolute time window"

time-window 60

when

\$throughput : Number(doubleValue < threshold) from
accumulate(Throughput(\$current : current),
average(\$current))

then

// raise the alarm

end

- Support separate time-windows per CE:

```
rule "absolute time window per CE"
```

```
when
```

```
    Stock( $id : id )
```

```
    Number( $monthAvg : doubleValue) from accumulate(  
        DayCloseStockPrice( id == $id, $value : value ),  
        average( $value ) ) over 30 days
```

```
    Number( intValue >= 3 ) from accumulate(  
        $o : SellOrder( completed == true, id == $id, value  
        < $monthAvg )  
        count( $o ) ) over today
```

```
then
```

```
    //sell stocks
```

```
end
```

- Reasoning over relative time windows
 - this is a powerful feature of reasoning over variable time-windows defined in relation to other patterns. Security example: "since a user logs in, until the user logs off, when there is any privileged action for this user, allow it and log it to the audit log"

rule "relative time window"

since

 LogInEvent(\$user : user)

until

 LogOffEvent(user == \$user)

when

 \$evt : Event(user == \$user, privileged == true)

then

 // log event

 // allow action

end

- Reasoning over relative time windows
 - "Each user has a limited amount he can buy/sell from the moment stock trading start to the moment stock trading stops, every day, without authorization. Every order that goes beyond that set amount (based on the user authorization level and trader policy) requires authorization to be completed."



rule "relative time window"

since

 StockTradeStartEvent(\$date : date)

until

 StockTradeStopEvent(date == \$date)

when

 User(\$user : user, \$level : level)

 TraderPolicy(level == \$level, \$max : maximumAllowance)

 Number(\$total : doubleValue) from accumulate(

 CompletedOrder(\$value : value),

 sum(\$value)

)

 \$o : OrderRequest(value > (\$max - \$total))

then

 // request authorization to complete order \$o

end

- Events have implicit time attributes and it must be possible to constraint events on its time attributes, using operators "after", "before", "between":

rule "time constraints between events"

when

 \$order : StockBuyOrder(\$id : id)

 StockBuyConfirmation(relatedEvent == id, this after [0,10])

then

 // buy order confirmed

end

- Support to reason over the absence of events:

rule "absence of events"

when

 \$temp : TemperatureReading(celciusGrade > \$threshold)

 not \$plinkerActivation(this after 10)

then

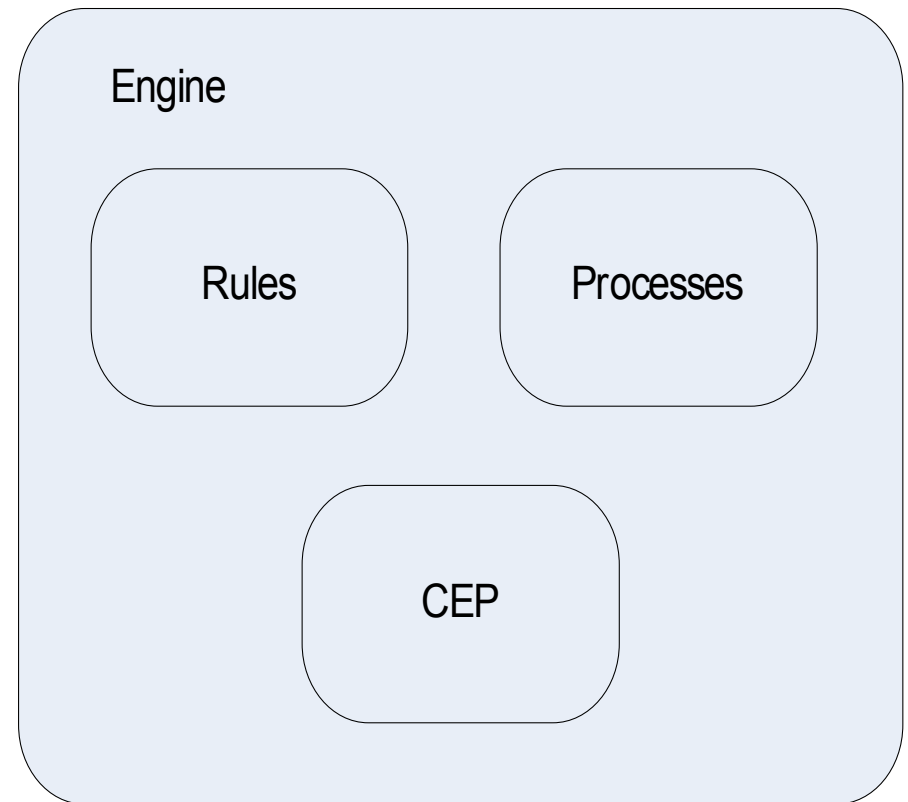
 // raise the alarm

end

■ Future

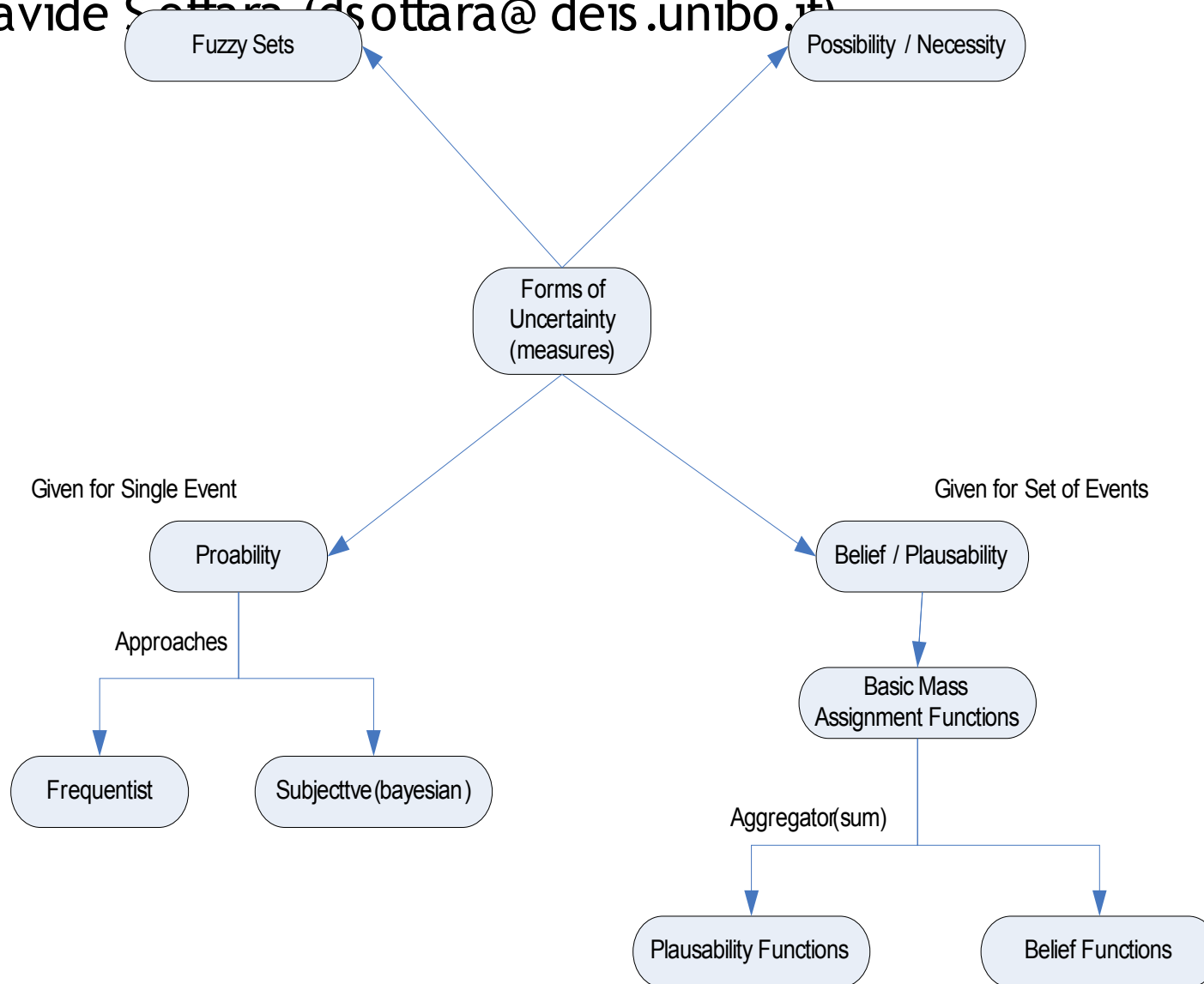
- More Complex forms of temporal reasoning
- More expressive forms of even object correlation
- Hybrid support for bayesian networks
- Scalability, High Availability
- ideas from the community? :)

- Engines and their applications emit events
 - Current node in process, time between nodes in process, number of executions of a rule, total rule executions in an engine, total number of cheeses bought.
- Rules, Processes and CEP
Are perfect for BAM
- Multi tier-ed (near/far) BAM
 - Internal self monitor (very near)
 - Local external monitoring (near)
 - Remote external Monitoring (far)

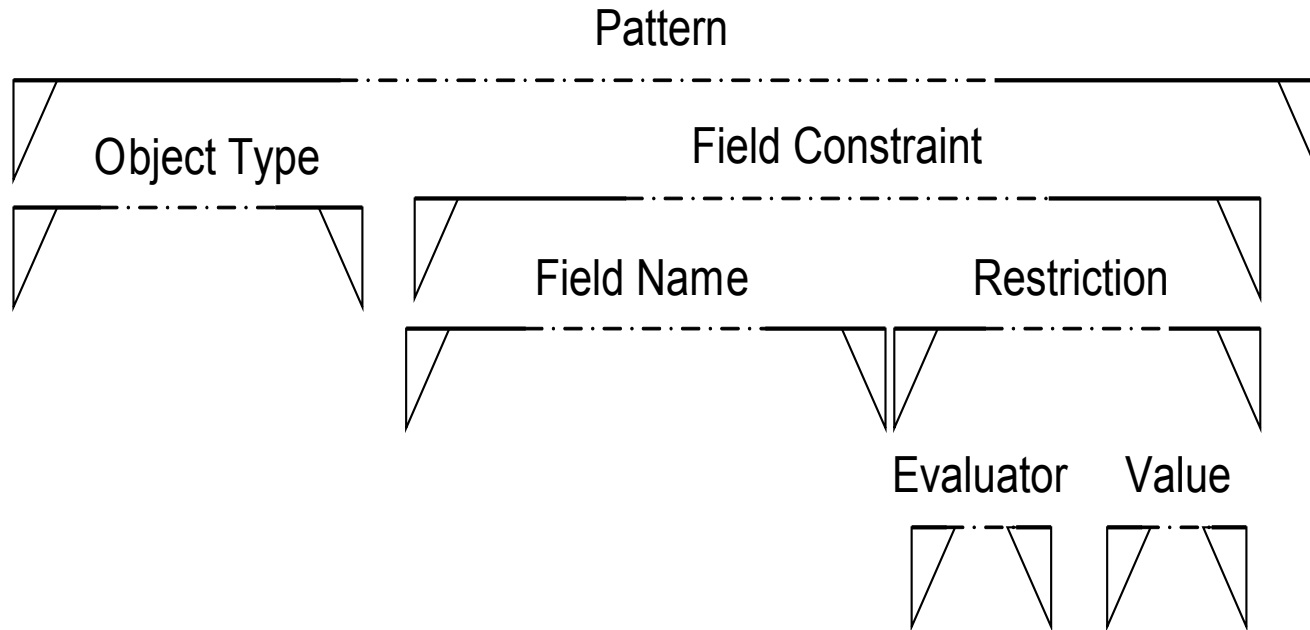


- OS Gi for the container
 - r-osgi (or similar)
 - SLP for discover and directory
 - Distributed OS Gi registries
 - p2p, p2m
 - FIPA messaging, instead of/as well as RMI?
 - High performance requires binary
 - XML adapters can be made for 3rd party support
- Leverage possible new CE
 - “from message”
 - Tuples enter the “message from” CE.
 - May optional send one or more messages
 - Incoming messages can join with one, a group or all Tuples

- Uncertainty Systems to express truth degrees and reason over partial data
 - Davide Sottara (dsottara@deis.unibo.it)



- Traditional Pattern
 - Shower(temperature == “hot”)



Shower(temperature == “hot”)

- Pattern with uncertainty evaluator
 - Shower(temperature == ~“hot”)
- Pattern with uncertainty evaluator and parameters
 - Shower(temperature == ~(10, \$x, 15, \$y) “hot”)

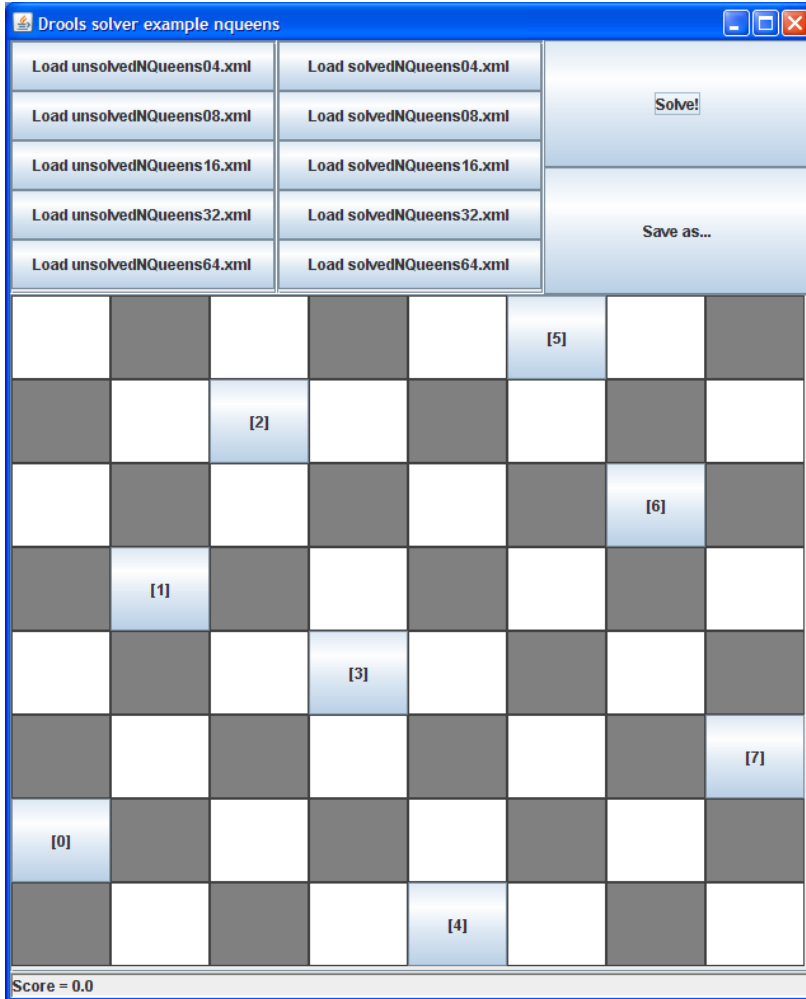
Drools-solver solves planning problems

Geoffrey De Smet

Drools-solver combines a search algorithm
with the power of the drools rule engine
to solve those planning problems

Solves planning problems, such as:

- Employee shift rostering
- Freight routing
- Supply sorting
- Lesson scheduling
- Exam scheduling
- The traveling salesmen problem
- The traveling tournament problem



- Place n queens that cannot attack each other on a $n \times n$ chessboard
- One of the examples of drools-solver
- Implementation explained in detail in the reference manual

Supports several search algorithms:

- Simple local search
- Tabu search
- Simulated annealing

You can easily switch the search algorithm, by simply changing the configuration.

```
<localSearchSolver>
  <scoreDriver> .../smartTravelingTournamentScoreRules.drl</scoreDriver>
  <scoreCalculator>
    <scoreCalculatorType>HARD_AND_SOFT_CONSTRAINTS</...>
  </scoreCalculator>
  <finish>
    <maximumMinutesSpent>2</maximumMinutesSpent>
  </finish>
  <selector> ...</selector>
  <accepter>
    <completeSolutionTabuSize>1500</completeSolutionTabuSize>
  </accepter>
  <forager>
    <foragerType>MAX_SCORE_OF_ALL</foragerType>
  </forager>
</localSearchSolver>
```

- Uses the drools rule engine to calculate the score
- Adding extra hard and soft constraints to the score function is **easy** and **scalable**

Example score rule

```
rule "fourConsecutiveHomeMatches"  
  when  
    $m : Match($homeTeam : homeTeam, $day1 : day)  
    Match(homeTeam == $homeTeam,  
          eval(day.getIndex() == $day1.getIndex() + 1))  
    Match(homeTeam == $homeTeam,  
          eval(day.getIndex() == $day1.getIndex() + 2))  
    Match(homeTeam == $homeTeam,  
          eval(day.getIndex() == $day1.getIndex() + 3))  
  then  
    // ...  
end
```


- More examples
- Refactor of the move generation (selector), to allow more efficient search algorithms
- Make moves declarative



- **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?