JBoss Rules – Viva Le Drools

Declarative Behavioural Modelling

An Integrated AI approach

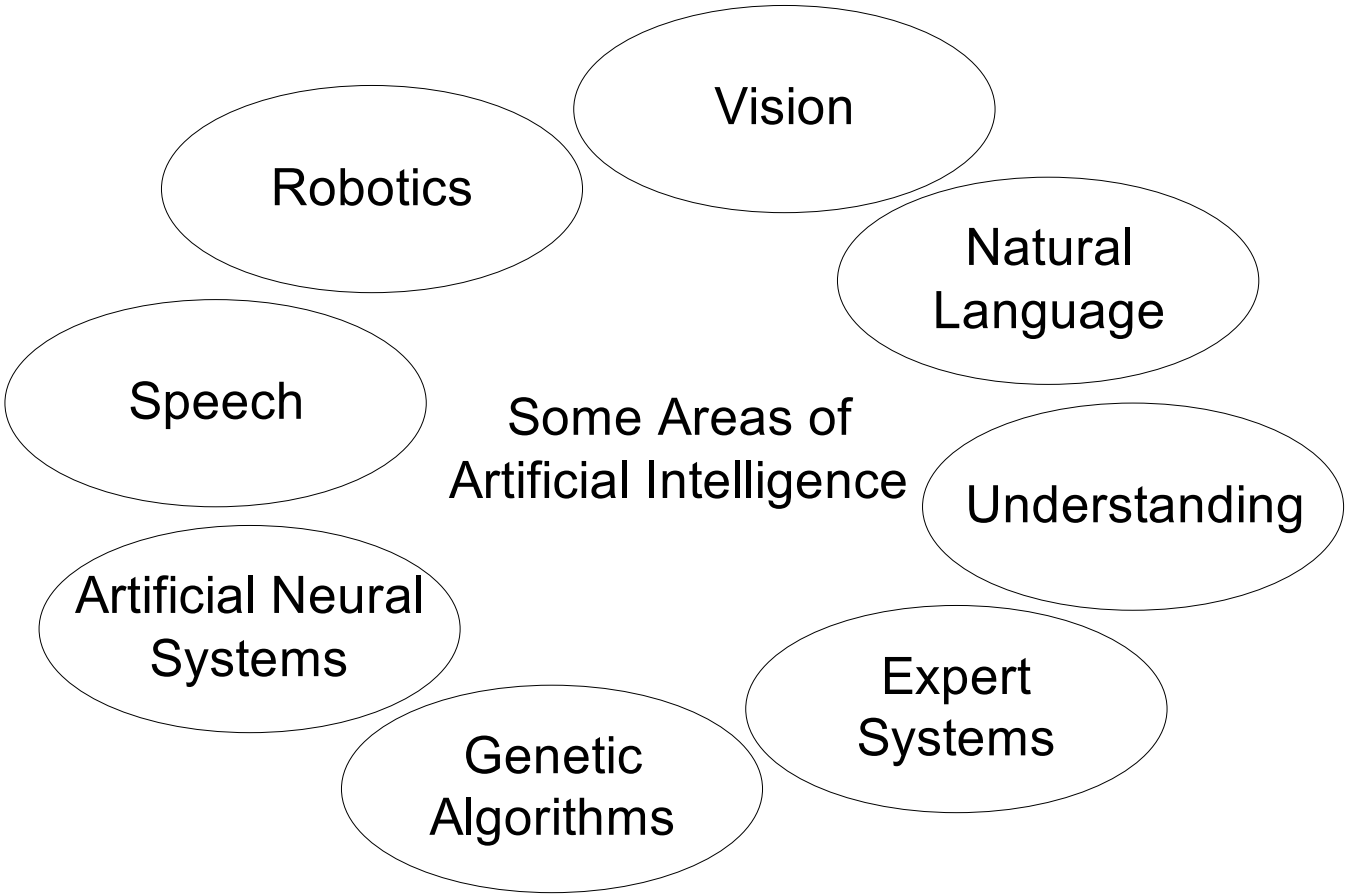
Mark Proctor

Project Lead

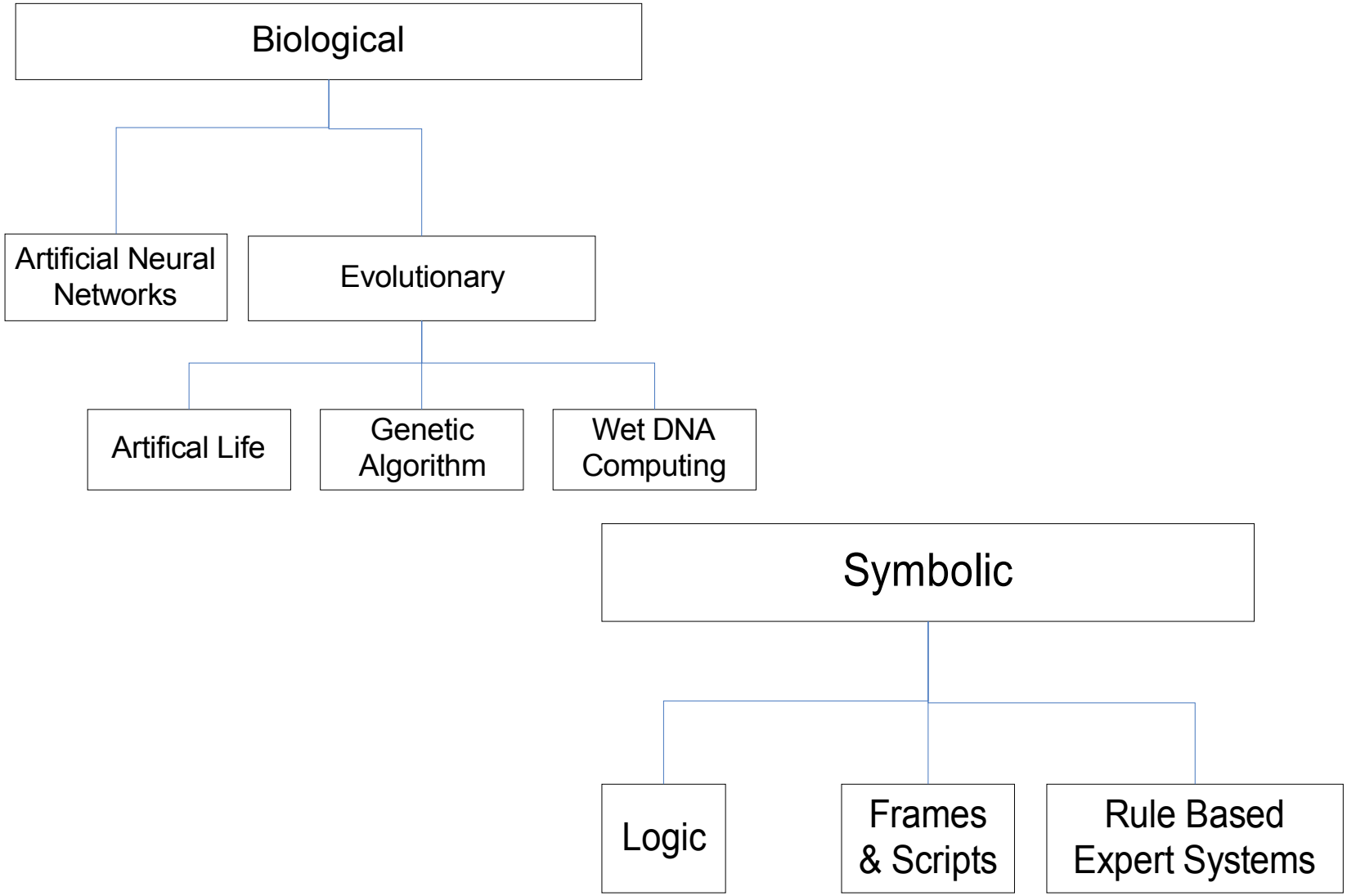
- The SkyNet funding bill is passed.
- The system goes online on August 4th, 1997.
- Human decisions are removed from strategic defense.
- SkyNet begins to learn at a geometric rate.
- It becomes self-aware at 2:14am Eastern time, August 29th
- In a panic, they try to pull the plug.
- And, Skynet fights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

Artificial Intelligence

Making computers think like people

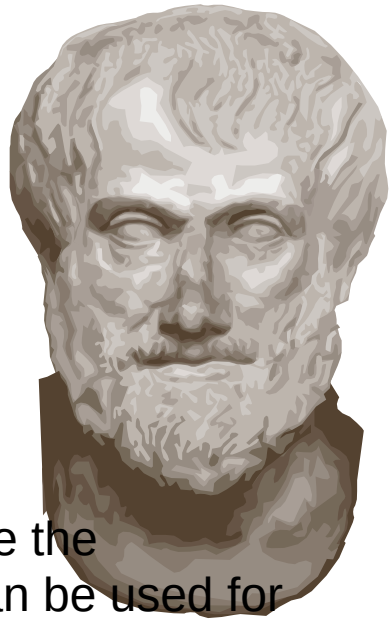


Branches of AI



Expert Systems - Knowledge Representation and Reasoning

- The study of Knowledge is Epistemology
- Nature, Structure and Origins of Knowledge



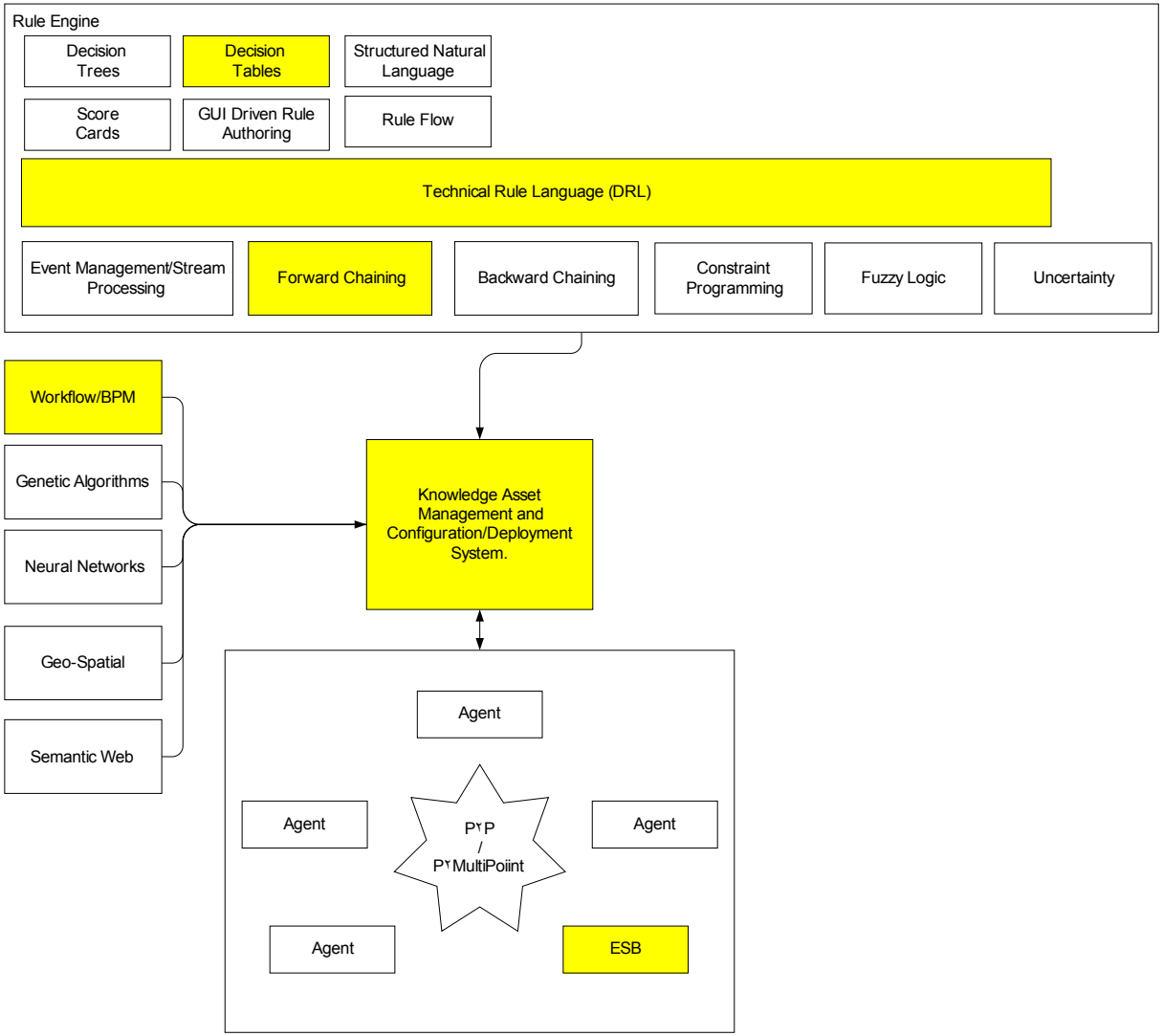
- Expert Systems use Knowledge representation to facilitate the codification of knowledge into a knowledge base which can be used for reasoning
  - we can process data with this knowledge base to infer conclusions

Production Rule System

- Turing Complete
  - Propositional Logic
  - First Order Logic
  - Declarative
- The Brain is the Inference Engine
  - scale to a large number of rules and facts
  - matches facts, the data, against Production Rules, also called Productions or just Rules, to infer conclusions which result in actions
  - A Production Rule is a two-part structure using First Order Logic for knowledge representation.

    when <conditions> then <actions>
  - The process of matching the new or existing facts against Production Rules is called Pattern Matching

Declarative Behavioural Modelling



The A-Team go Shopping

Team		
name	**role**	**rank**
Hannibal	Leader	Colonel
Faceman	Treasurer	Lieutenant
B.A.	Mechanic	Sergeant
Murdoch	Pilot	Captain

At Anne Summers

It's playtime! Dressing up outfits for him and her from Ann Summers - Mozilla Firefox

http://www.annsummers.com/products.asp?gid=6&cat=3

Welcome to Ann Summers

SPECIAL OFFER — FREE P&P! ON ALL ORDERS £45 OR OVER

Search the site

Type in search Go

- sex toys
- dressing up
- lingerie
- fun stuff
- oils &lotions
- hen night
- books & DVDs
- bondage
- AS uncut
- gift ideas
- for him
- knickerbox
- bingo
- SALE

CLICK HERE TO RETURN TO HOME PAGE

dressing up & playtime

Sister of Mercy | Miss Massage (BEST SELLER) | Schoolgirl (NEW!) | Air Hostess

Stop and Search | Naughty Nurse | Barrack Babe | Miss Bunny

Hail Mary | Miss Hot Cakes | She Devil | Tululah

Top Sex Toys

- Thruster Rabbit
- NEW! Love Bug
- The Cone
- Vibro Ring
- Promises
- Sinnflut

Top 5 New Products

- Rampant Rabbit Thruster
- Rampant Rabbit Platinum
- Rampant Rabbit Thriller
- Vibro Ring
- Original Blue Pill

Basket

Basket items: 0

Value: £0.00

View Basket

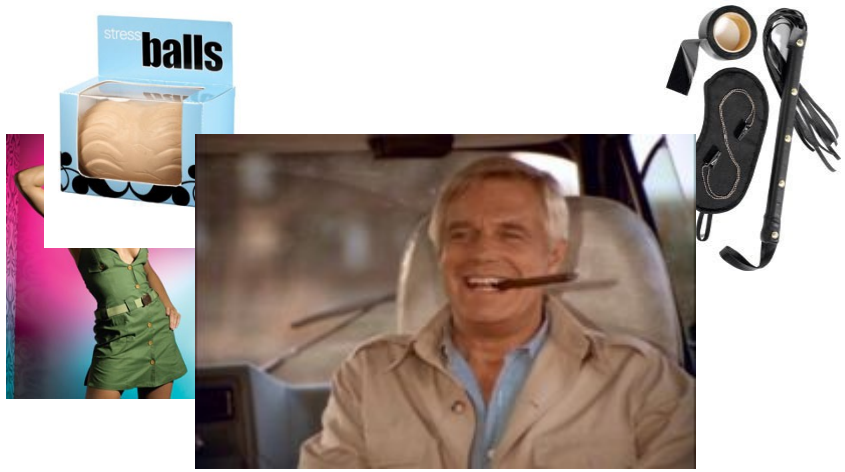
Proceed to Checkout

also on this site...

- home
- New! Advert
- AS on your mobile
- kate lawler
- news and gossip
- sex tips
- ann summers parties
- freedom parties
- ann summers stores
- order a catalogue
- get our emails
- contact us
- jobs @ ann summers
- become an affiliate
- jacqueline gold
- david gold
- david gold's book

Done

What do they Buy?



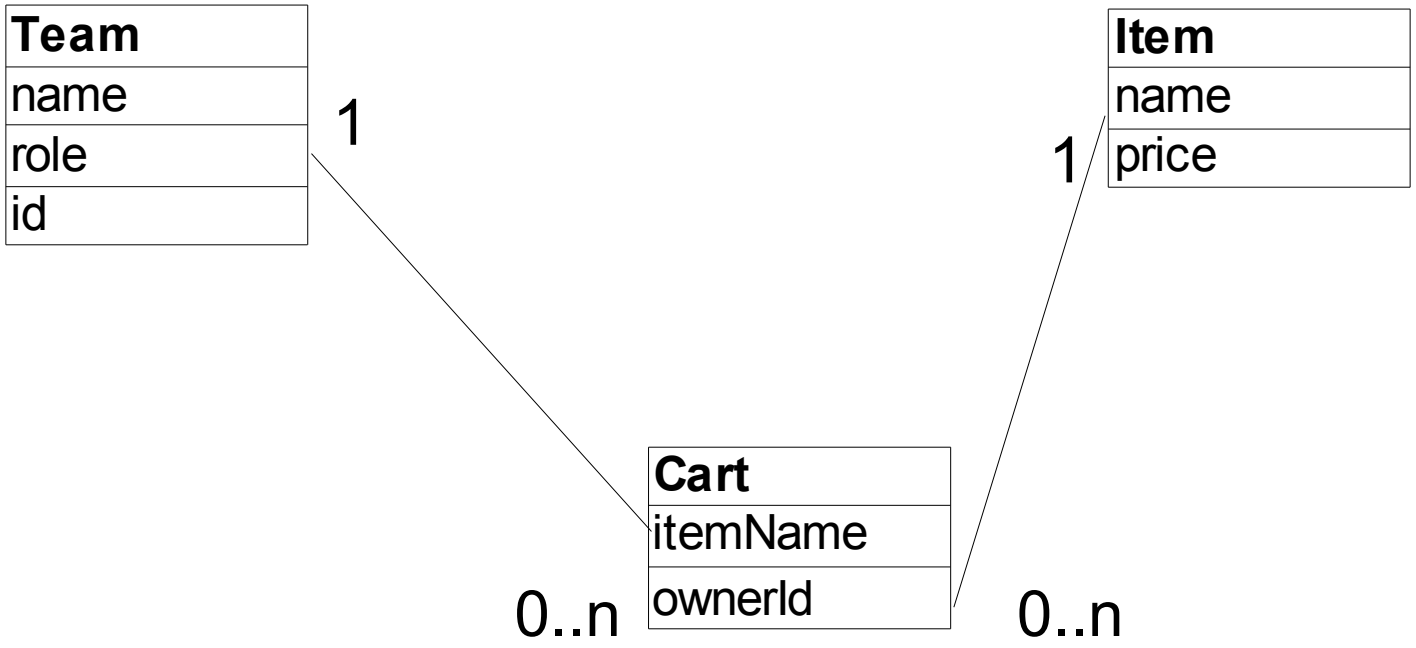
Tables

Team			
name	**id**	**role**	**rank**
Hannibal	٠	Leader	Colonel
Faceman	١	Treasurer	Lieutenant
B.A.	٢	Mechanic	Sergeant
Murdoch	٣	Pilot	Captain

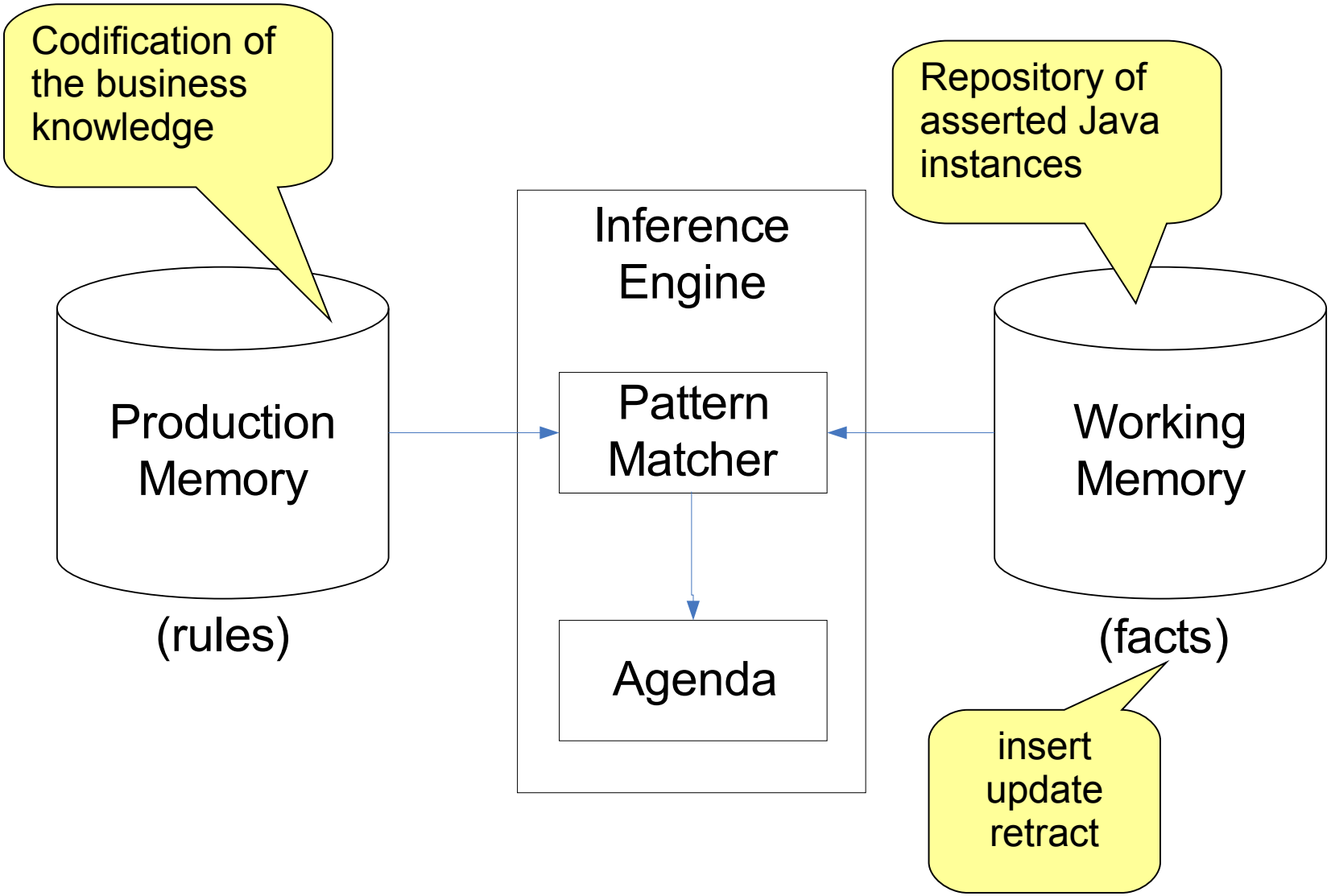
Barrack Babe	٠
Fur Love Cuffs	٢
Love Swing	٢
Studed Wristband	٢
Bondage Bear	٣
Bondage Starter Kit	٠
Bondage Starter Kit	١
Bondage Starter Kit	٢
Nymphette basque	١
Stress Balls	١
Stress Balls	٠
Chocolote Body Paint	٣

Item	
name	**price**
Barrack Babe	٢٥
Fur Love Cuffs	٨
Love Swing	٢٠٠
Studed Wristband	٥
Bondage Bear	٤
Bondage Starter Kit	٨
Nymphette basque	٢٥
Stress Balls	٦
Chocolote Body Paint	٥

Relationships



What is a Production Rule System



What is a Rule

Quotes on Rule names are optional if the rule name has no spaces.

- ```
  rule "<name>"
      <attribute> <value>
      when
          <LHS>
      then
          <RHS>
  end
  ```

salience	<int>
agenda-group	<string>
no-loop	<boolean>
auto-focus	<boolean>
duration	<long>

RHS can be any valid java. Future versions will support other languages, i.e Groovy
  ```

What is a Rule

Methods that must be called directly

specific passing of instances

- ```
  public void helloMark(Person person) {
      if ( person.getName().equals( “mark” ) {
          System.out.println( “Hello Mark” );
      }
  ```

Rules can never be called directly

Specific instances cannot be passed.

LHS

- ```
  rule “Hello Mark”
      when
          Person( name == “mark” )
      then
          System.out.println( “Hello Mark” );
  end
  ```

RHS

Package

Namespace for all package members

Imports can be used in functions and rules. Uses valid java import syntax

```
package com.sample

import java.util.Map
import com.sample.Cheese

global Cheese cheese

function void exampleFunction(Cheese cheese) {
    System.out.println( cheese );
}

rule “A Cheesy Rule”
    when
        ….
    then
        ….
end
```

Expressiveness

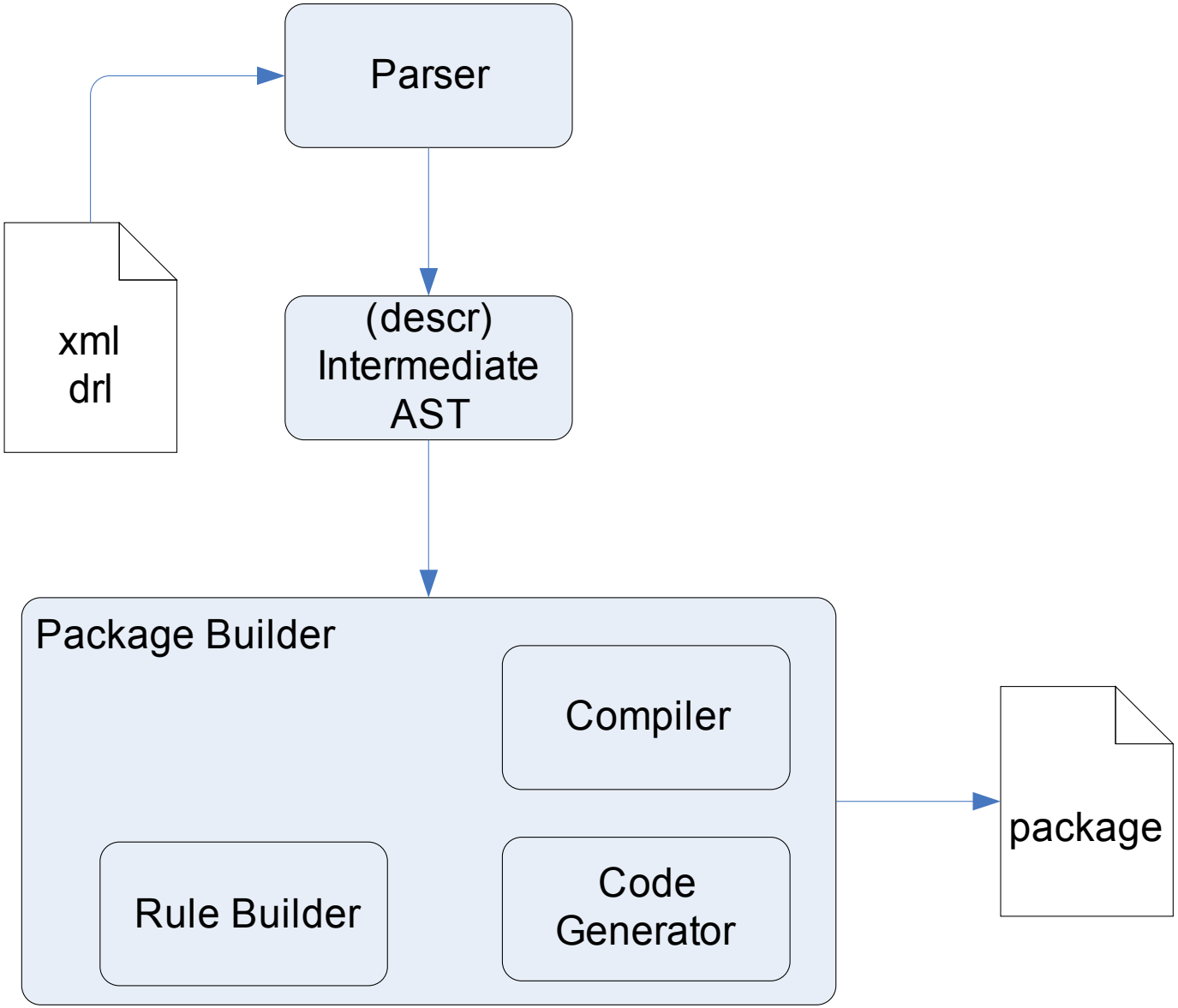
- **Turing Complete**
  - Propositional Logic
  - First Order Logic
- **Propositional Logic**
  - Cheese.name == “stilton”
- **First Order Logic (Quantifiers)**
  - Exists
  - Not
  - Accumulate
  - Collect
  - From
  - Forall
- **Execution Control**
  - Conflict Resolution (salience)
  - Agenda Groups
  - Activation Groups
  - Rule Flow
- **Temporal Rules**
  - Scheduler

Expressiveness

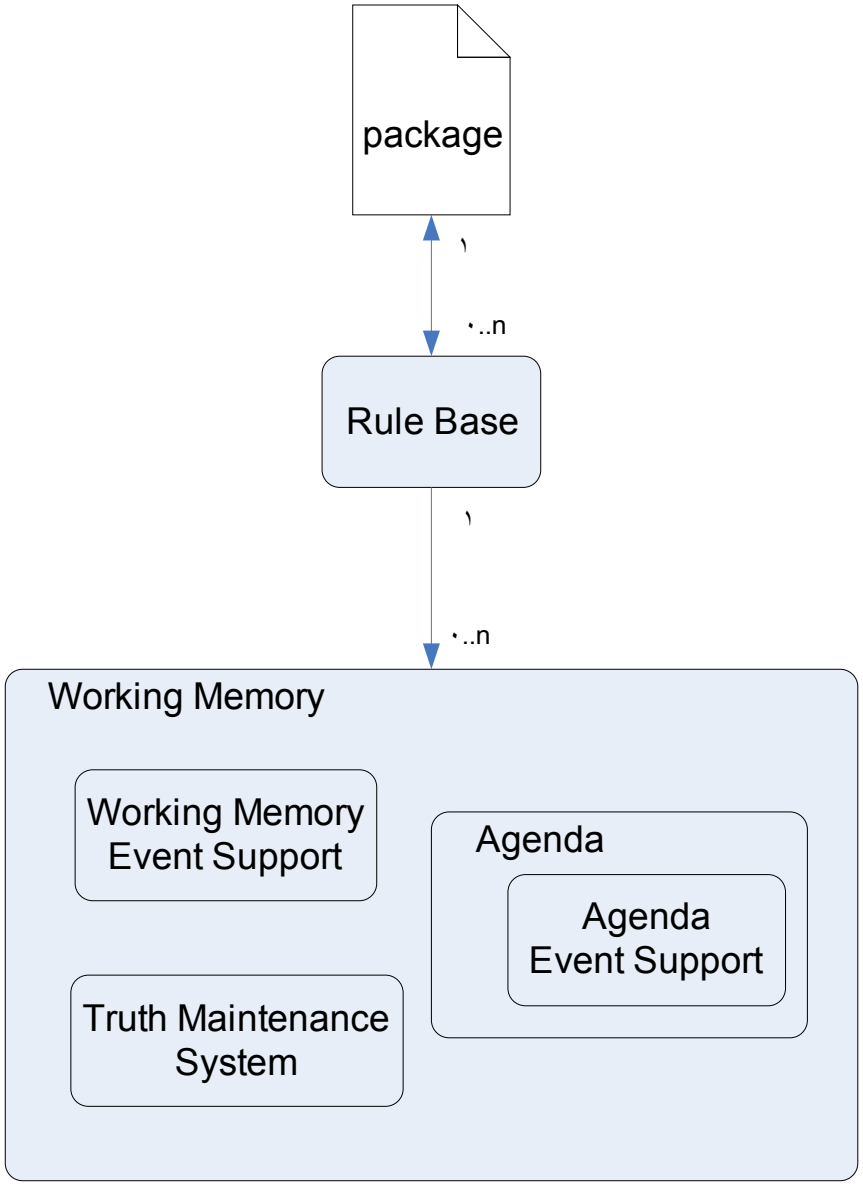
- **Truth Maintenance**
  - Logical Objects
  - Compensating Actions/Rollbacks (todo)
- **Nesting of conditional elements inside quantifiers**

- **Backward chaining**
- **Uncertainty**
  - Bayesian Logic
  - Fuzzy Logic
- **Event Stream/Management Processing**
- **Constraint Programming (solver)**

Authoring API



Runtime API


package
1
0..n
Rule Base
1
0..n
Working Memory
Working Memory Event Support
Agenda
Agenda Event Support
Truth Maintenance System

Object Insertion and Pattern Matching

- **LHS**
  - One or more Patterns
  - Patterns are the conditions that must be satisfied for the rule to be legible for firing
- **Object assertion**
  - Patterns within the Rule Base are matched. Resulting in partial and full matches for Rules.
  - Fully matched Rules result in the creation of an Activation
  - No rules fire at this stage

Object Modification

- **How to modify a object in the Working Memory**
  - From Java Code

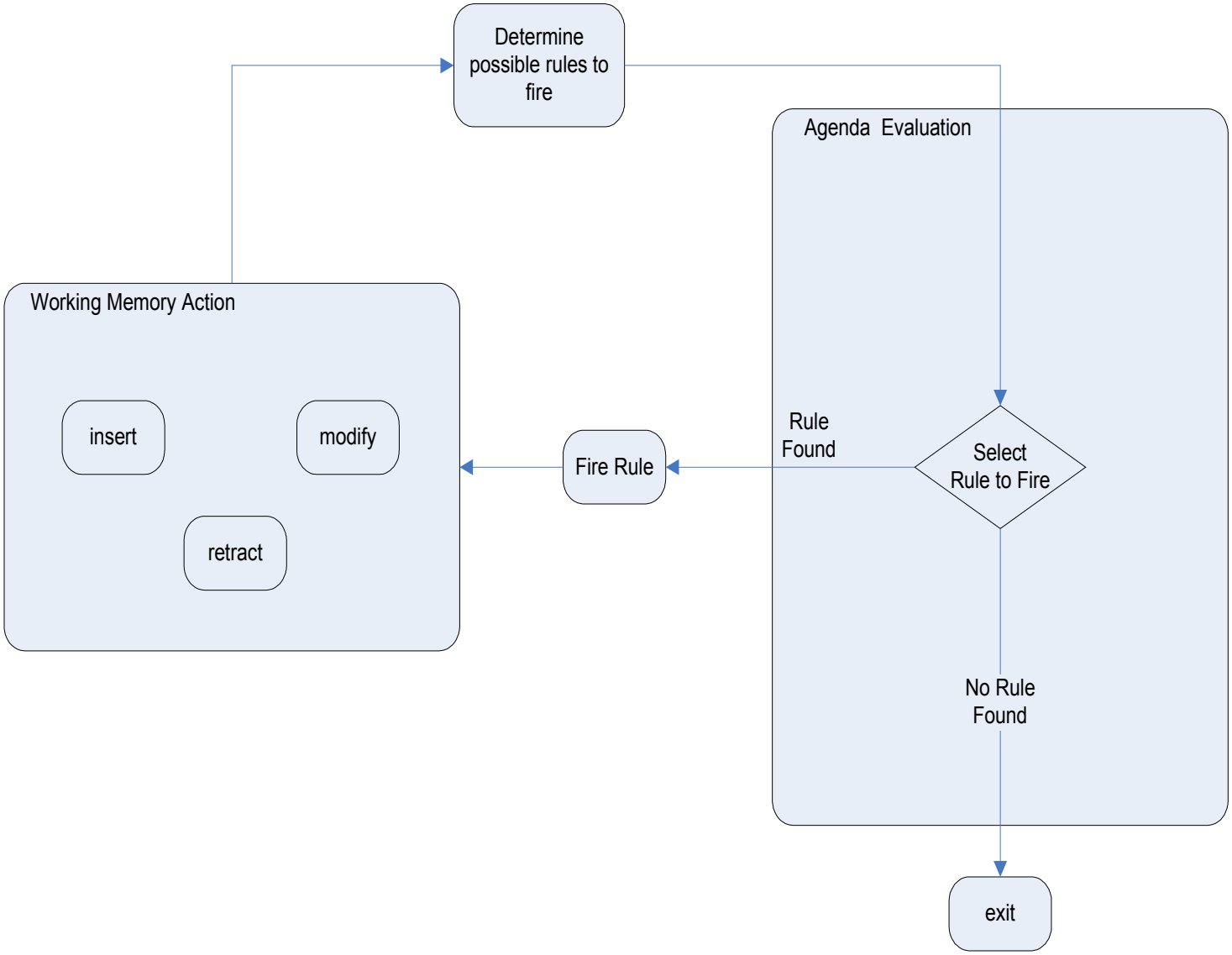
    workingMemory.update(factHandle, modifiedFact)
  - From a Consequence

    update(modifiedFact)
- **JavaBeans PropertyChangeListeners can provide automatic notification.**
- **Modifications result in**
  - Activation Cancellations
  - Activation Creations
  - Internally this is similar to a retract and assert

Two Phase System

- **Working Memory Actions**
  - Occurs in Java code and during the execution of a Consequence
  - Assertion
  - Deletion
  - Modification
- **Agenda Evaluation**
  - Triggered by Calling workingMemory.fireAllRules()
  - Executes the first Rule's Consequence and enters Working Memory Action phase. At the end of the Consequence it returns to evaluating the Agenda.
  - When the Agenda is empty it returns back to the main Java code.

Two Phase System



Working With Objects

```
Item
    String name;

Customer
    int id
    Item[] cart
```

```
Customer customer = new Customer( "Fred Flinstone" );
customer.addItem( new Item( "brie" ) )
customer.addItem( new Item( "cheddar" ) )
customer.addItem( new Item( "feta" ) )
workingMemory.insert( customer )
```

```
rule "Message the customers who have not bought any brie"
    when
        $customer : Customer( $cart : cart -> ( ! $cart.includes(
new Item( "brie" ) ) ) )

    then
        $customer.sendMessage( "Brie is your best
friend" );
end
```

More Expression

- **3.0.x only allows comma seperated field constraints. 'or' could be used at the CE level, but resulted in subrule generation.**
  - Can now use && and || inside the pattern for multiple values on the same field and across files – no subrule generation.
  - Person(age > 30 && < 40 || hair =="black")
- **3.0.x auto-have autovivification of variables in dialect expressions**
  - Before: Cheese(oldPrice : oldPrice, newPrice == (oldPrice * 1.10))
  - Now: Cheese(newPrice == (oldPrice * 1.10))

More Expression

- **3.0.x had to always declare the variable, causing cluter, can now access direct properties of pattern variables.**
  - Before: p : Person(personId : id)
    i : Item(id == personId, value > 100)
  - Now: p : Person()
    i : Item(id == p.id, value > 100)
- **Eval rewrite for complex expressions**
  - Before: Person($pets:pets
    eval($pets['rover'].type == "dog")
  - Now: Person(pets['rover'].type == "dog")

Rule Engines are Relational

```
Customer
    int id

Item
    int customerId
    String name
```

```
Customer customer = new Customer( "Fred Flinstone" );
workingMemory.insert( customer );
workingMemory.insert( new Item( "brie", customer.getId() )
workingMemory.insert( new Item( "cheddar", customer.getId() )
workingMemory.insert( new Item( "feta", customer.getId() )

rule "Message the customers who have not bought any brie"
    when
        Customer( $customerId : id )
        not ( Item( customerId == $customerId, name ==
"brie" ) )

    then
        $customer.sendMessage( "Brie is your best
```

Exploiting Relational Data in 3.2

- 'forall'
- ‘from’
- ‘collect’
- ‘accumulate’

- **Forall**
  - True when the pattern is true for all facts
  - Forall(Bus(color == “red”))
- **From**
  - Pulls and unifies against none working memory data

Can call hibernate querries

Sub fields

```
Restaurant( rating == “five star” )
        from hbSession.getNamedQuery( “restaurant query” ).
            setProperties( key1 : value1, key2 : value2).list()
```

‘from’

```
rule "Message the customers who have not bought any brie, and
haven't bought brie in previous shopping trips"
    when
        Customer( $customerId : id )
        not ( Item( customerId == $customerId,
                    name == "brie" ) )

         not ( Item() from hibernateSession.getNamedQuery( "How
        much cheese?" ).setProperties( { customerId =
        $customerId,
                                          type => "brie" ) )
    then
            $customer.sendMessage( "You really haven't had
    enough Brie recently, remember Brie is your best
    friend" );
    end
```

Collect

- **Collect**
  - Allows you to use cardinality
  - When there are more than 6 red buses
  - List(size > 6) from collect (Bus(color == “red”))
  - 'from' can be chained. Following is true if all items in a cart have a price creater than 10
  - List(size == ($list.size)) from collect(Item(price > 10)
    from $cart.items

‘collect’

```
rule "If we continuously have less than 10 brie items, then
do a  discount"
    duration 60000 //1 minute
    when
        $context : Conext( count < 10 )
        cheeseList  : ArrayList(size < 10)
                      from collect Item( name == "brie" )
    then
        $context.setCount( $context.getCount() + 1 );
end
```

```
rule "If we continuously have less than 10 brie items, then
do a  discount"
    duration 60000 //1 minute
    when
        $context : Conext( count < 10 )
        cheeseList  : ArrayList(size > 10)
                      from collect Item( name == "brie" )
    then
        $context.reset();
end
```

‘collect’

```
rule "If we continously have less than 10 brie items, then
do a  discount"
    duration 60000 //1 minute
    when
        $context : Conext( count >= 10 )
        cheeseList  : ArrayList(size < 10) from collect
Item( name == "brie" )
    then
        // do discount
end
```

Accumulate

- **Accumulate**
  - More powerful 'collect' allows you to execute actions on each matched fact in the set
  - $total : Integer()

```
    from accumulate( $item : Item( )
                     init(count = 0; total=0)
                     action(count++;total += $item.price)
                     result( return total/count )
```

‘accumulate’

```
 duration 60000 //1 minute
    when
        Integer( intValue < 3 )
        from accumulate( Item( name == "brie",
                               $weight : weight ),
                         init( int totalWeight = 0, count; ),
                         action( count++;
                                 totalWeight += $weight; ),
                                 result( new Integer( x ) ) );

    then
        // do discount
end
```

Line Debugger and new Rete Viewer

Debug - StateExampleUsingSalience.drl - Eclipse SDK

File Edit Navigate Search Project Run Window Help

Debug

StateExampleUsingSalience [Drools Application]
org.drools.examples.StateExampleUsingSalience 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

Variables | Breakpoints

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

StateExampleUsingSalience.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" );
```

Text Editor | Rete Tree

Properties

Property	Value
General	
Constraint	[LiteralConstraint fieldExtra
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

Con... | Sea...

StateExampleUsingSalience [Drools

A finished

Global Data View

The selected working memory has no globals defined.

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

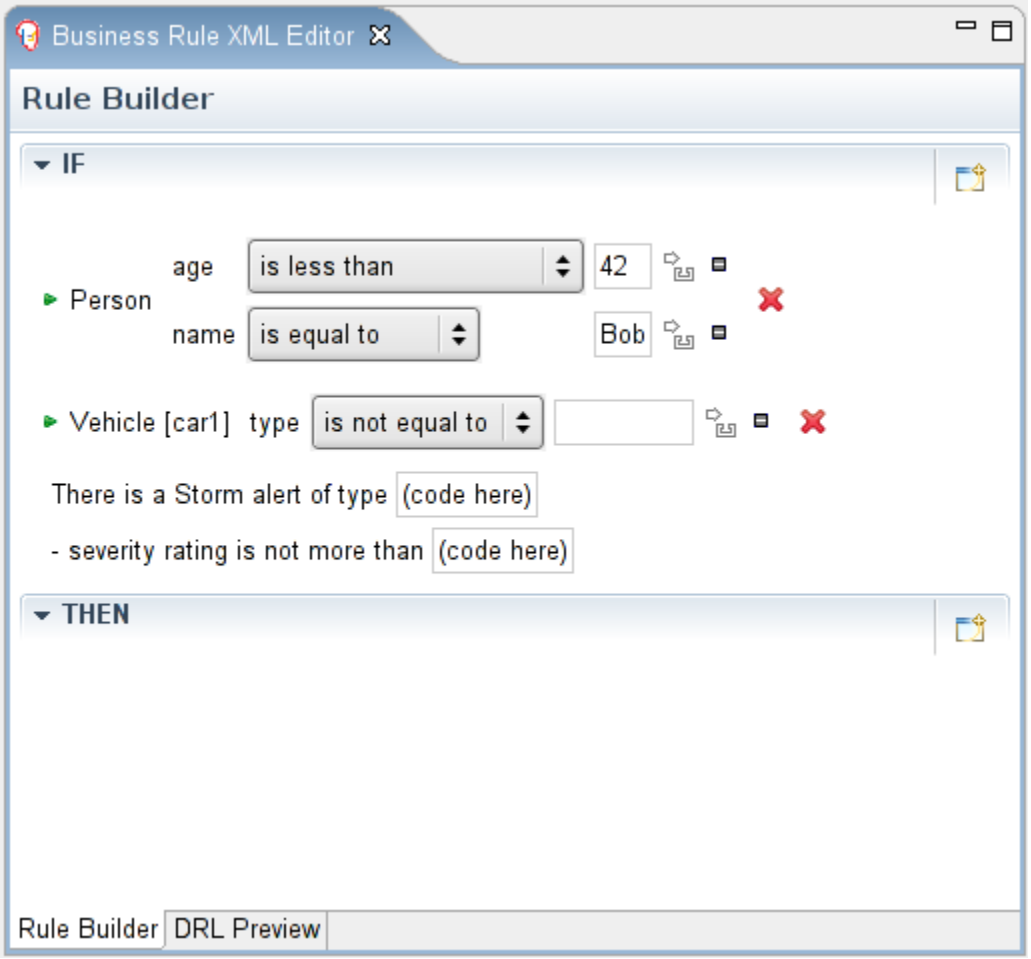
MAIN[focus]= AgendaGroupImpl (id=1259)
[0]= AgendaItem (id=1262)
ruleName= "B to C"
c= State (id=1269)
[1]= AgendaItem (id=1263)
ruleName= "B to D"
d= State (id=1270)

Working Memory View

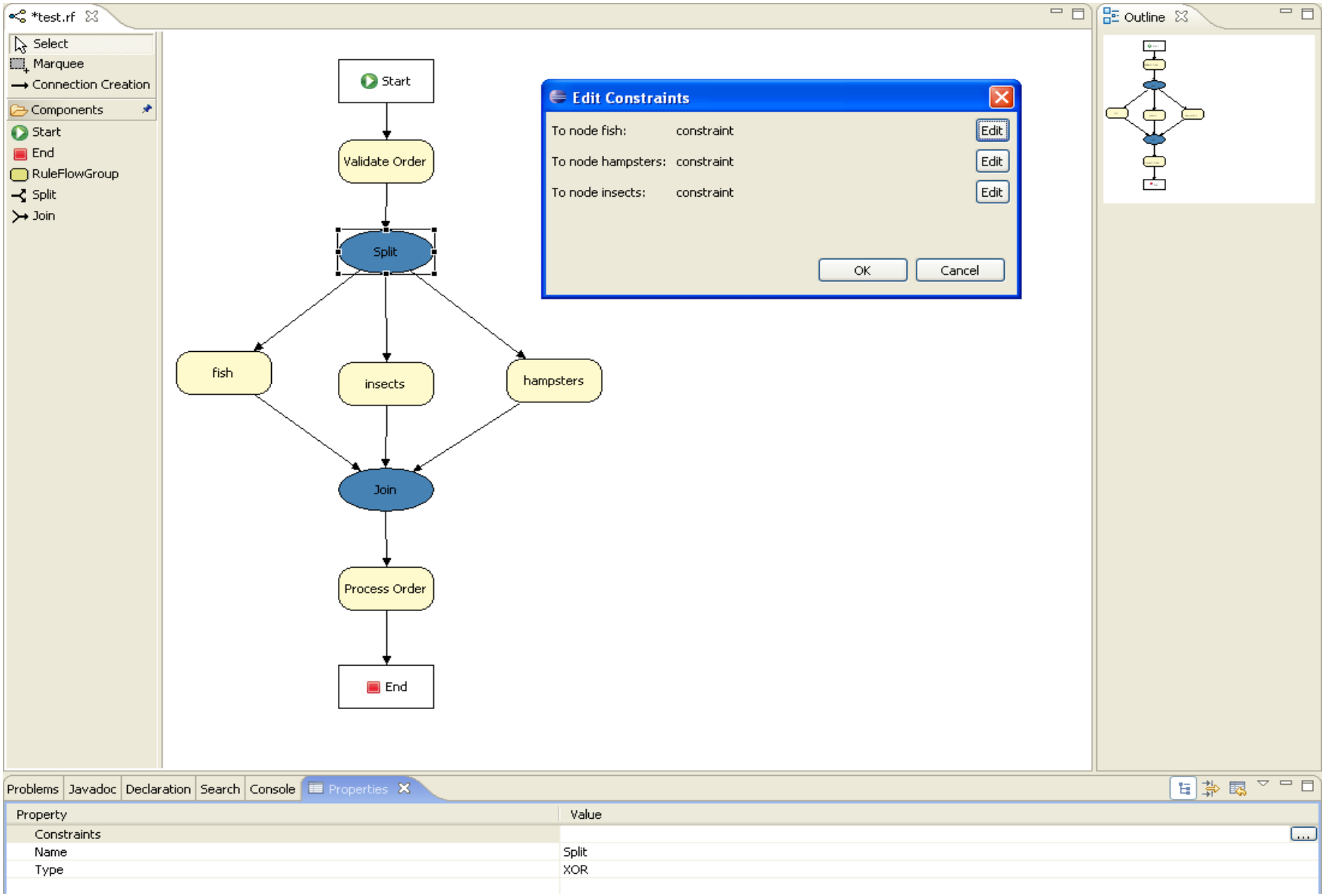
[0]= State (id=1268)
[1]= State (id=1269)
FINISHED= 1
NOTRUN= 0
changes= PropertyChangeSupport (id=1294)
name= "C"
state= 0
[2]= State (id=1270)
[3]= State (id=1271)

Writable | Insert | 21 : 1

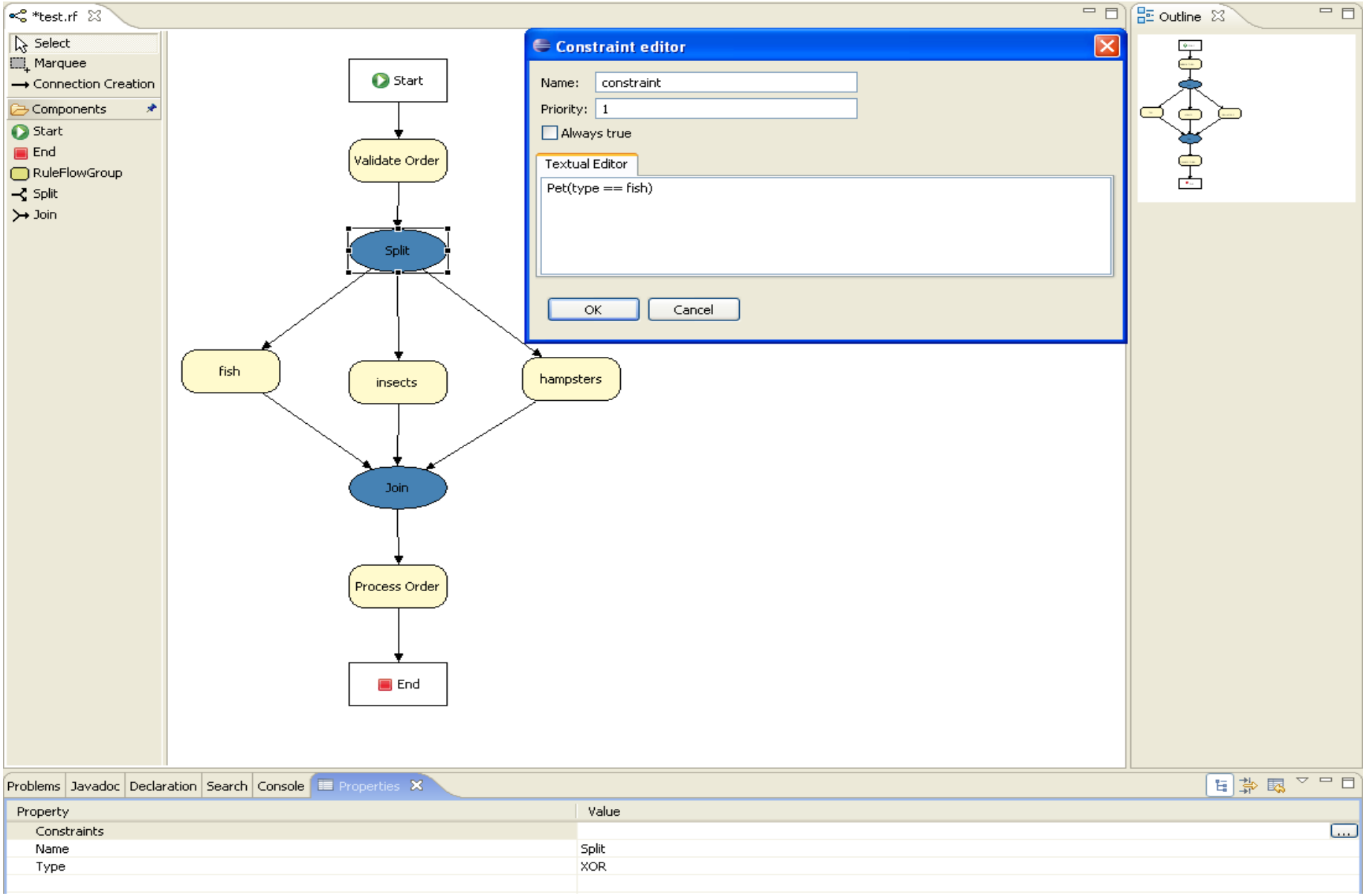
Eclipse Guided Editor



Rule Flow


*test.rf
Select
Marquee
Connection Creation
Components
Start
End
RuleFlowGroup
Split
Join
Start
Validate Order
Split
fish
insects
hampsters
Join
Process Order
End
Edit Constraints
To node fish: constraint
Edit
To node hampsters: constraint
Edit
To node insects: constraint
Edit
OK
Cancel
Outline
Problems
Javadoc
Declaration
Search
Console
Properties
Property
Value
Constraints
Name
Split
Type
XOR

Rule Flow


*test.rf
Select
Marquee
Connection Creation
Components
Start
End
RuleFlowGroup
Split
Join
Start
Validate Order
Split
fish
insects
hampsters
Join
Process Order
End
Constraint editor
Name:
constraint
Priority:
1
Always true
Textual Editor
Pet(type == fish)
OK
Cancel
Outline
Problems
Javadoc
Declaration
Search
Console
Properties
Property
Value
Constraints
Name
Split
Type
XOR

Pluggeable Dialects

- **Return-value, predicate, evals and consequences can now specify dialects, now suppors Java and MVEL .**
  - Cheese(type == "stilton",
    eval(price == (new Integer(5) + 5)),
    price == (new Integer(5) + 5))
  - Assert (new Person()) (name = “mark”, age = 31);

Why MVEL

- **Reflection/bytecode(JIT) compilation and execution modes.**
  - For huge systems we need to be able to avoid excessive bytecode generation, but still have the option for bytecode JIT for performance sensitive areas.
- **Fast reflection mode.**
  - We originally started with our own language JFDI, which was designed to be a simple and fast reflection based language, the idea is all work is done at compile time so runtime is just a series of reflection invokers. This design has been carried through to MVEL, so that it has good enough reflection performance. Where as other languages have to drop reflection mode and use bytecode to get any reasonable level of performance.
- **Pluggeable resolvers.**
  - Dictionary population is too slow, MVEL can resolve it's variable direct from the provided resolvers, which we make array based for performance.
- **Size.**
  - MVEL is currently <>

Why MVEL

- **Custom language extensions.**
  - MVEL is extending the language to support rule friendly constructs, in particular block setters. So I can do "modify (person) (age += 1, location = "london")" with the ability to treat that as a transaction block so I can run before and after interceptors on the entire block. This is made easier through the use of macros, so we can define our own keywords and have them expanded into mvel code.
- **Static/Inferred typed or dynamic modes.**
  - Variables can be untyped and totally dynamic.
  - Variables can be statically typed or type can be inferred, casting is supported.
  - Optional verifier for "typed mode", disallows dynamic variables and ensures all types and method calls are correct. Which helps with.

    Authoring time validation.

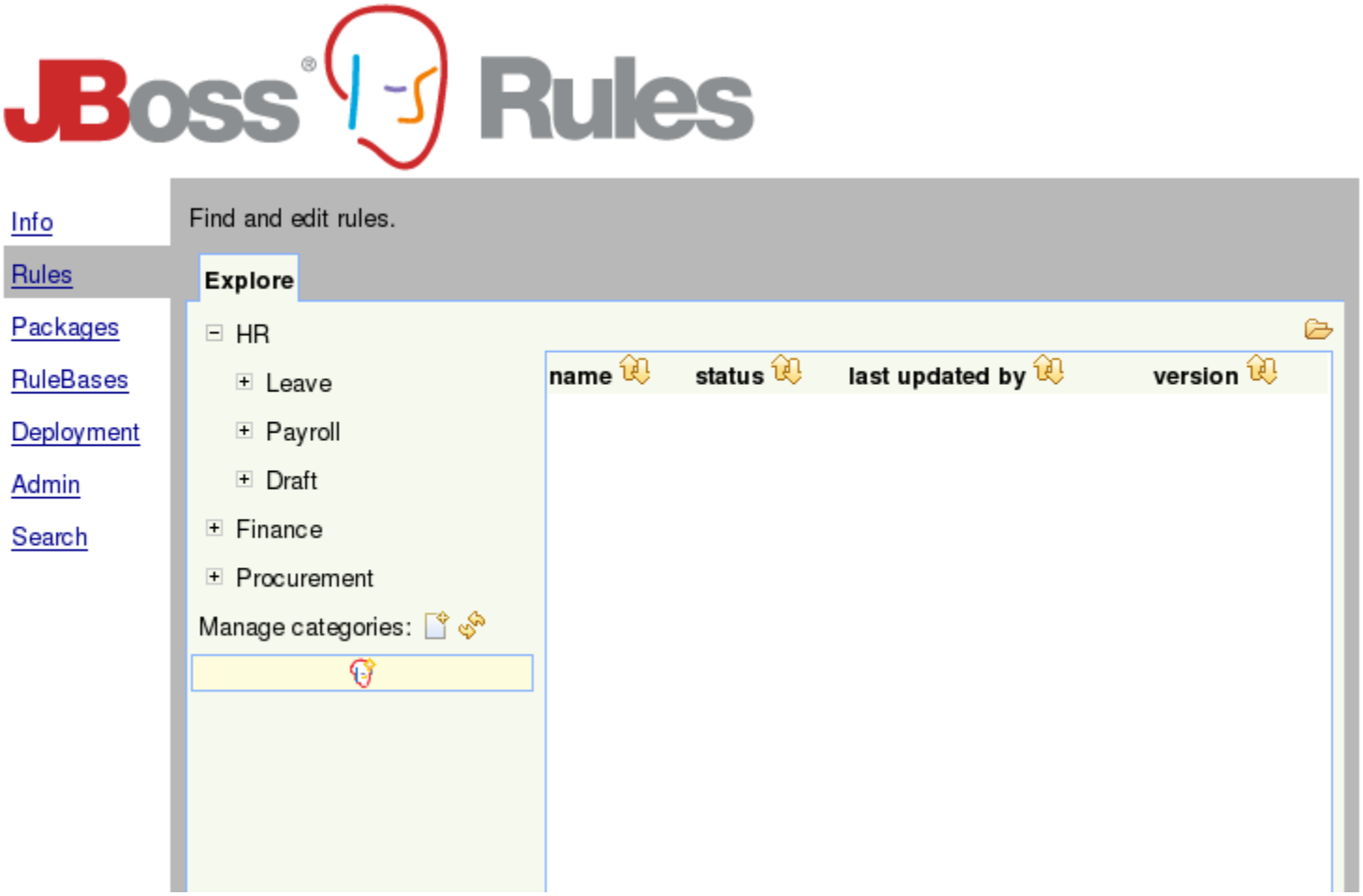
    Code completion.

    Refactoring.
- **Configurable language feature support.**
  - Language features can be turned off.
  - We don't want imperative flow structures in the "then" part, no 'if' 'switch' etc. Rules should be declarative, "when this do that" not "when this maybe do that".

BRMS?

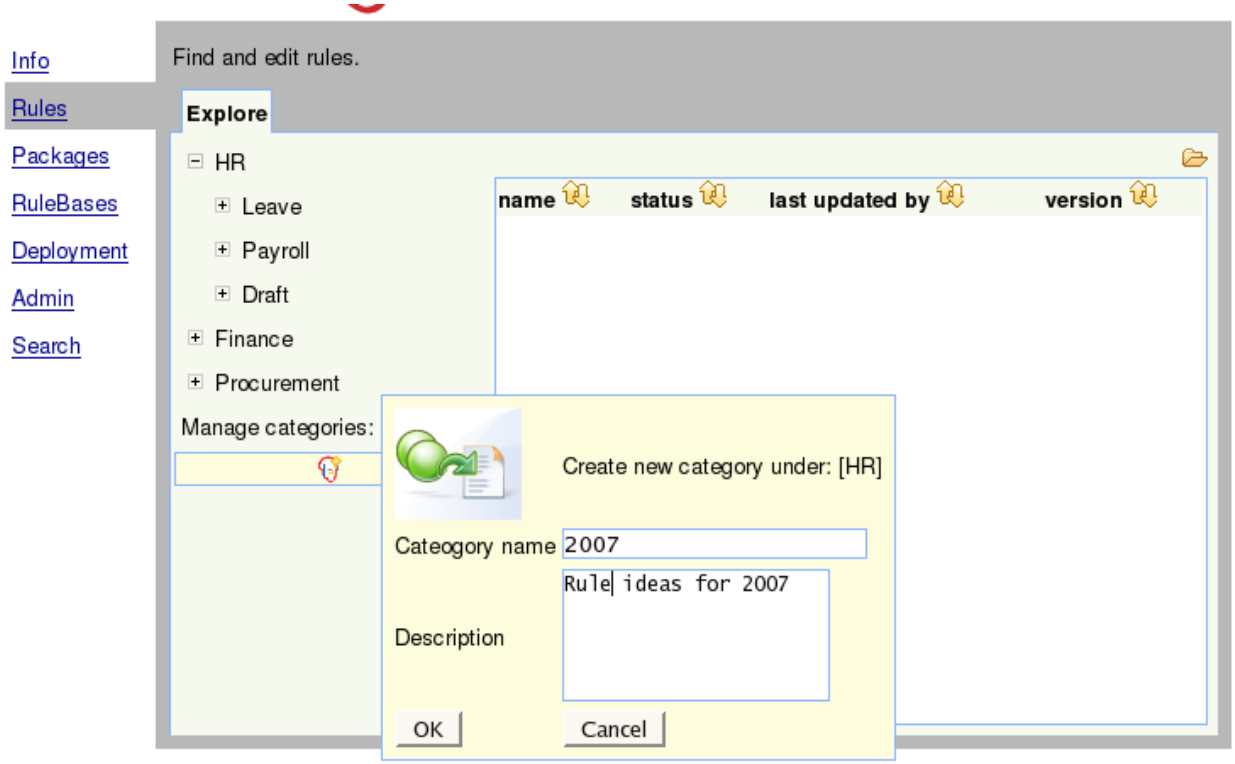
- Business Rules Management System
- Why?
  - **For managing a whole enterprises declarative rules**
  - **(and knowlege assets)**
  - **eg 5000 + rules for mortgage pricing**
  - **Business focussed view, not developer focussed**
  - **Versioning, editing, validating, FINDING (!), approving, searching, controlling, auditing, XXX-ing.**
- Needs to complement developer tools, NOT REPLACE

Rule explorer with categorisation



Categorisation of assets is critical

- Its how you find stuff
- Categories are completely user/business driven

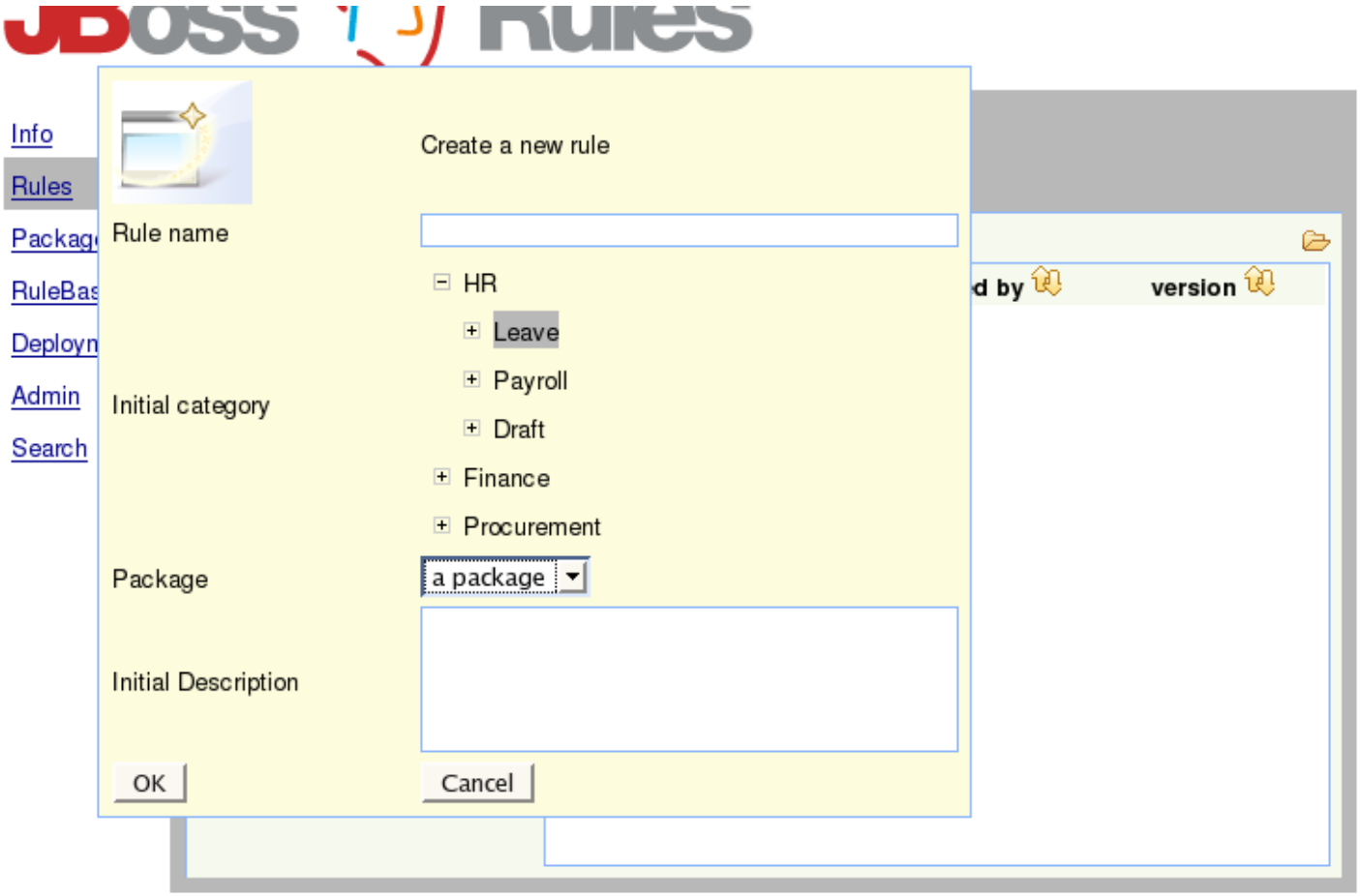


Dublin Core

- Encourage structured classification
- Future archeologists may be able to make sense of it ;)
- Its a prescriptive set of attributes to attach to an asset

Business friendly rules

- **Controlled rule creation, authoring**



Versioning

- **We developers take it for granted**
- **Its good**
- **Business Analysts need it**
  - But they have manually manage their requirements/rules documents
  - Have a manual workflow
  - Have manual versioning
  - No body knows the horrors I have seen

Friendly rule editing

Info
Rules
Packages
RuleBases
Deployment
Admin
Search

Find and edit rules.

Explore | name 6

name 6

Categories: HR, Finance, Procurement

Subject: age rejection for unlinked

Last modified on:

Last modified by:

Created by:

Version number: 0

Package:

Type: rejection

External link: RJ1024

Source: QLD MOTR ACT 2003

IF

There is a Driver

- age less than age years old

THEN

Reject Policy with explanation : ' reason '

According to QLD MOTR Act 2003, we cannot insure this age range of drivers according to our current licence.

They will need to have a "guarantor" type relationship with a previously approved customer (we can give the customer a discount too, as a consolation prize !).

BRMS

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 – JBos... | Google Calendar | Red Hat -- Intranet H... | Nopaste | Brisbane Times – New...

JBoss Rules

Info
Rules
Packages
Deployment
Admin

Find and edit rules.

Explore | Rule_2 | MyDT | Rule_1

Status: [Draft] Save changes | Copy | Archive | Delete

MyDT

Upload new version: Browse... Upload

Download current version: Download

This is a decision table in a spreadsheet (XLS). Typically they contain many rules in one sheet.

View source | Validate

<documentation>

MyDT

Categories: HR

Modified on: Fri 11 May 2007 06:04:21 PM EST
by:
Note: Initial
Version: *Not checked in yet*
Created on: Fri 11 May 2007 06:04:21 PM EST
Created by: default
Format: xls

Package: DemoPackage
Subject:
Type:
External link:
Source:

Version history

Done

BRMS

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 – JBos... | Google Calendar | Red Hat -- Intranet H... | Nopaste | Brisbane Times – New...

JBoss Rules

Info
Rules
Packages
Deployment
Admin

Find and edit rules.

Explore | Rule_2 | MyDT | Rule_1

Status: [Draft] Save changes | Copy | Archive

IF

Person

age less than or equal to 42

age greater than 21

Board [b]

Board

There is no 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

Done

BRMS

JBoss Business Rules Management System

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

User: fmeyer [Sign Out]

Info
Rules
Packages
Deployment
Admin

Administer the repository

Manage categories | Manage stat

Create a new top level category.

Cateogory name: Fibonacci

Description: fibonacci category

OK | Cancel

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

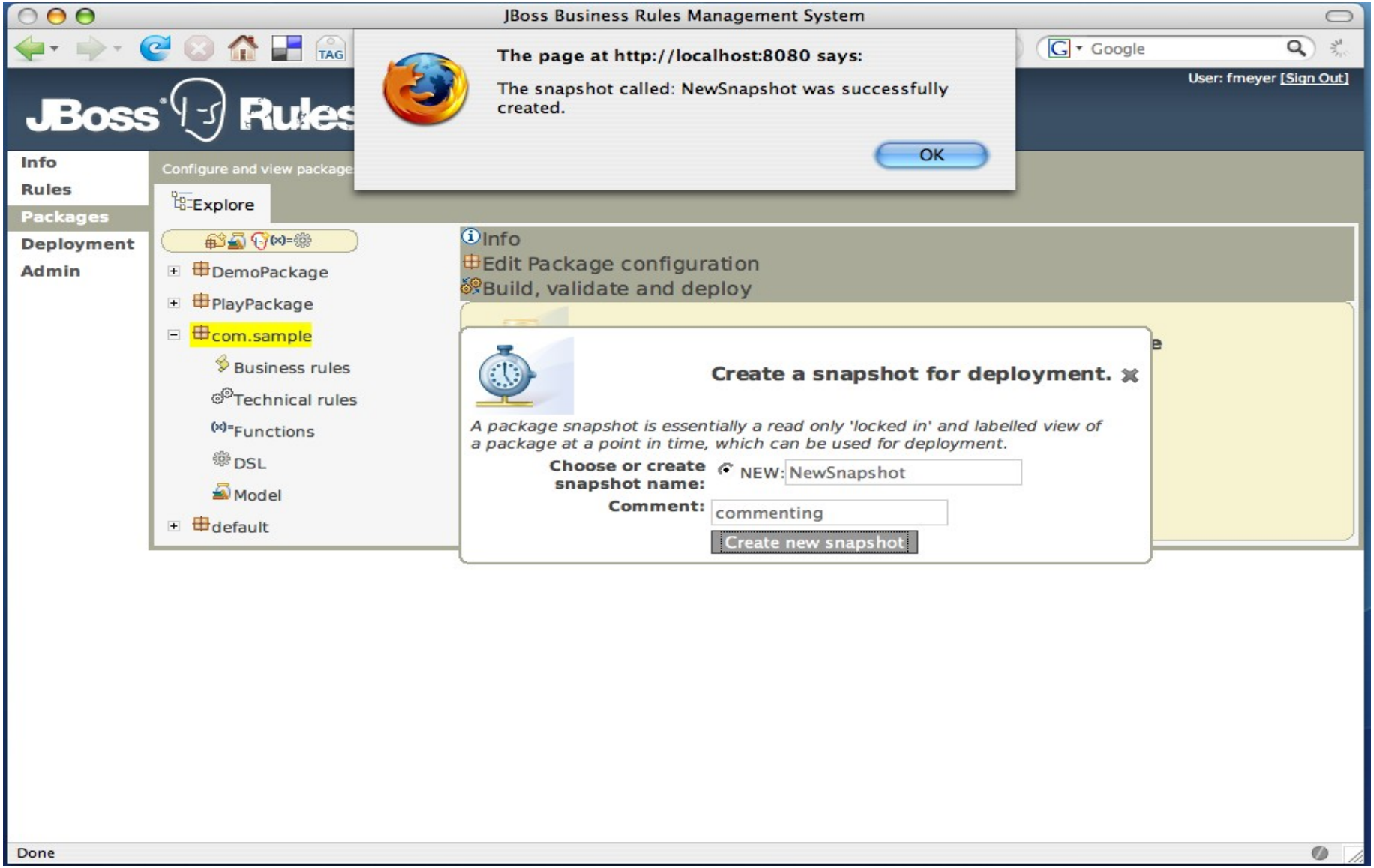
Current categories:
- HR
- Finance
- Draft

Refresh view:

Create a new category:

Delete the currently selected category:

BRMS


JBoss Business Rules Management System
The page at http://localhost:8080 says:
The snapshot called: NewSnapshot was successfully created.
OK
User: fmeyer [Sign Out]
Info
Rules
Packages
Deployment
Admin
Configure and view packages
Explore
DemoPackage
PlayPackage
com.sample
Business rules
Technical rules
Functions
DSL
Model
default
Info
Edit Package configuration
Build, validate and deploy
Create a snapshot for deployment.
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.
Choose or create snapshot name:
NEW: NewSnapshot
Comment:
commenting
Create new snapshot
Done

BRMS

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 – JBos... | 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
```

View source | Validate

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

Done

Technical versus business rules

- **A powerful inference engine allows you to solve hard problems**
- **Not ALL of the hard problem is technical**
  - Thats the “business” part of the rules
- **For the parts that are technical, use the technical rule language**
  - “Make the easy parts declarative, and the hard parts procedural”

Technology involved

- **BRMS “client” is a web app**
- **Ajax via GWT**
- **JCR (Jackrabbit default implementation)**
  - popular standard for content management

Questions?

- 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?**