



# Drools Expert & Fusion



## What a rule-based program is

Made up of discrete rules, each of them models a specific aspect of your domain  
 Simpler because you can concentrate on the rules for one situation at time  
 Flexible in the face of fragmentary or poorly conditioned inputs

**Declarative** *Vs.* **Imperative**  
 (What to do) *Vs.* (How to do it)

## Features

Hybrid reasoning  
 Forward and Backward Chaining  
 Reactive, Query and Materialized Views  
 Functional Programming  
 Truth Maintenance  
 Temporal Reasoning  
 Complex Event Processing  
 Calendar and Timer Support  
 Dynamic Object Classification (Traits)

## Examples

```
rule "Check age" when
    $a : Applicant( age < 18 )
then
    modify( $a ) { valid = false }
end

rule "Acknowledgement Failure" when
    $req : BuyReq()
    not BuyAck( this after[15s] $req )
then
    channel["alerts"].send( new Alert( $req ) );
end

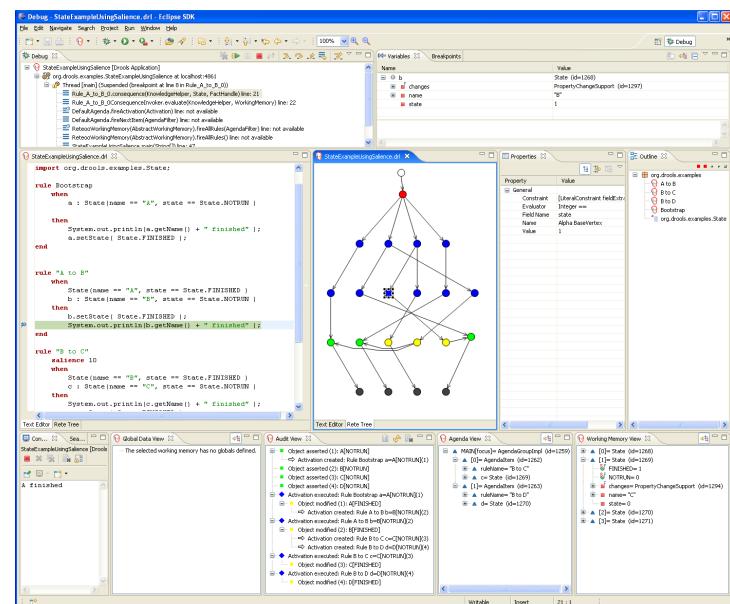
rule "Buy More Stock" when
    TradingWindow( status == OPEN )
    Accumulate( $t : Ticker( name == "RHT" ) over window:time(4m),
                $avg : avg( $t.price ) )
    eval( $avg > 100 )
then
    channel["buy channel"].send( new BuyReq( "RHT" ) );
end
```

## Complex Event Processing with Drools Fusion

Understands and handles events as a first class platform citizen (specialized Facts)  
 Selects a set of events in a cloud and detect the relevant relationships among them  
 Takes appropriate action based on the patterns detected and the events' temporal relationships

## When should you use Drools?

The problem is beyond any algorithmic solution or isn't fully understood  
 The business-logic changes often  
 Domain expert or business analysts are readily available but are nontechnical  
 You want to isolate the key parts of your logic especially *the really messy parts*



# Drools Integration

## ... with Spring

Spring is a popular framework for enterprise Java development.

Drools ships with a Spring-ready module that allows Drools to be configured and deployed into the Spring container without any glue code.

For more details on spring, visit:

<http://www.springsource.org>

## Sample Configuration - Camel

Example 1.1. Drools EndPoint configured with the CXFRS producer

```
<bean id="droolsPolicy" class="org.drools.camel.component.DroolsPolicy" />

<camelContext id="camel" xmlns="http://camel.apache.org/schema/spring">
    <route>
        <from uri="cxfrs://bean://rsServer"/>
        <policy ref="droolsPolicy">
            <unmarshal ref="xstream" />
            <to uri="drools:node1/ksession1" />
            <marshal ref="xstream" />
        </policy>
    </route>
</camelContext>
```

## Sample Configuration - Spring

Example 2.2. kbase definition example

```
<drools:kbase id="kbase1" node="node1">
    <drools:resources>
        <drools:resource source="classpath:org/drools/spring/IntegrationExampleTest.xls"
                        type="DTABLE">
            <drools:decisiontable-conf input-type="XLS" worksheet-name="Table_2" />
        </drools:resource>
        <drools:resource ref="resource1"/>
        <drools:resource source="classpath:org/drools/container/spring/model.xsd" />
    </drools:resources>
    <drools:configuration>
        <drools:mbeans enabled="true" />
        <drools:accumulate-functions>
            <drools:accumulate-function name="func1" ref="func1Instance" />
            <drools:accumulate-function name="func1" ref="func2Instance" />
        </drools:accumulate-functions>
    </drools:configuration>
</drools:kbase>
```

## Drools Server

The Drools Execution Server (drools-server) module is a war, which can be deployed in a application server (such as JBoss AS), and allows applications to execute KnowledgeBases remotely for any sort of client application. This is not limited to JVM application clients, but any technology that can use HTTP, through a REST interface.

The execution server leverages both Drools Camel and Drools Spring modules to support stateless and stateful sessions in a native way.

Check the Drools Integration documentation for more information:

[www.jboss.org/drools/documentation.html](http://www.jboss.org/drools/documentation.html)

## ... with Apache Camel

Apache Camel is a powerful open source integration framework based on known Enterprise Integration Patterns.

Drools ships with an Apache Camel endpoint implementation for out of the box integration with hundreds of other products that also integrate through Apache Camel.

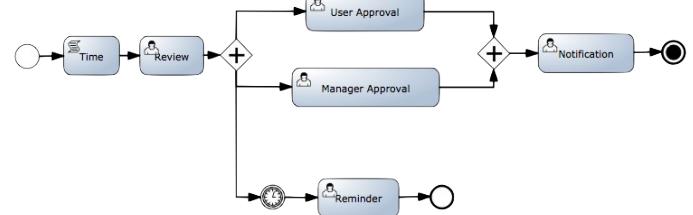
For a list of Apache Camel components, visit:

<http://camel.apache.org/>



**Open-source business process management project offering:**

- generic process engine supporting native BPMN 2.0 execution
- targeting developers and business users
- collaboration, management and monitoring using web-based consoles
- powerful rules and event integration



<b>Core Engine</b>		
Core engine is a workflow engine in pure Java		
<ul style="list-style-type: none"> <li>- Lightweight</li> <li>- Embedded</li> <li>- Generic</li> <li>- Extensible</li> </ul>		
<b>BPMN2</b>		
<b>OMG Specification</b>		
<ul style="list-style-type: none"> <li>- Model, Notation, Execution</li> <li>- Understandable</li> <li>- Collaboration and Choreography</li> </ul>		
<b>Persistence and Transactions</b>		
<b>Persistence (JPA, pluggable)</b>		
<ul style="list-style-type: none"> <li>- Runtime Persistence</li> <li>- History logging</li> <li>- Services</li> </ul>		
<b>Transactions (JTA, pluggable)</b>		
<ul style="list-style-type: none"> <li>- Command scoped</li> <li>- User-defined</li> </ul>		
<b>Console</b>		
<b>Web-based management</b>		
<ul style="list-style-type: none"> <li>- Targets Business Users</li> </ul>		
<b>Features</b>		
<ul style="list-style-type: none"> <li>- Manage process instances</li> <li>- User task lists / forms</li> <li>- Reporting</li> </ul>		
<b>Integration</b>		
<b>Domain-Specific processes</b>		
<ul style="list-style-type: none"> <li>- Extend palette with custom, declarative service nodes</li> </ul>		
<b>Human-Task Service (WS-HT)</b>		
<ul style="list-style-type: none"> <li>- Task Lists, Task Lifecycle</li> </ul>		
<b>Task Clients</b>		
<ul style="list-style-type: none"> <li>- Task form generation</li> </ul>		

- **Manage process instances:** start new process instances, inspect the state of currently running instances (status window and

- **Design:** create/update your business process via graphical representation.
- **Test and Debug:** test specific process scenarios and temporarily halt process executions to inspect their current state
- **Simulate:** visualize process progress, manipulate the execution clock, send timers, validate assertions at specific execution points, and

- **Design:** create and update your business processes in the Web-based environment
- **Validate:** validate your business processes visually
- **Generate:** generate fully executable process and task forms. Generate your process images and view your process in numerous formats
- **Connect:** connect to the jBPM Service Repository to install numerous domain-specific service nodes
- **Migrate:** migrate your existing jBPM 3 process definitions to BPMN2 with a single click

# Guvnor – Decision Tables

## Types

### Limited Entry

Body contains boolean states indicating which constraints or actions apply

### Extended Entry

Body contains domain values

min-age [18]	max-age [25]	Minimal cover	Maximum cover	BMW	318i	M3	premium [1000]	premium [1500]	premium [2000]
Applicant [Sa]			Policy [Sp]		Vehicle [Sv]			\$p	\$p
age [≥18]	age [≤25]	type [=TPFT]	type [=COMP]	make [=BMW]	model [=318i]	model [=M3]	premium [1000]	premium [1500]	premium [2000]
+	+	✓	✓	□	✓	✓	□	✓	□
+	+	✓	✓	□	✓	□	✓	□	✓
+	+	✓	✓	□	✓	✓	□	□	□
+	+	✓	✓	□	✓	□	✓	□	□

#	Description	min-age	max-age	policy type	make	model	premium
		Applicant [Sa]	Policy [Sp]	Vehicle [Sv]			\$p
		age [≥]	age [≤]	type [=]	make [=]	model [=]	premium
+		1	18	25	TPFT	BMW	318i
+		2	18	25	COMP	BMW	318i
+		3	18	25	TPFT	BMW	M3
+		4	18	25	COMP	BMW	M3
							2500

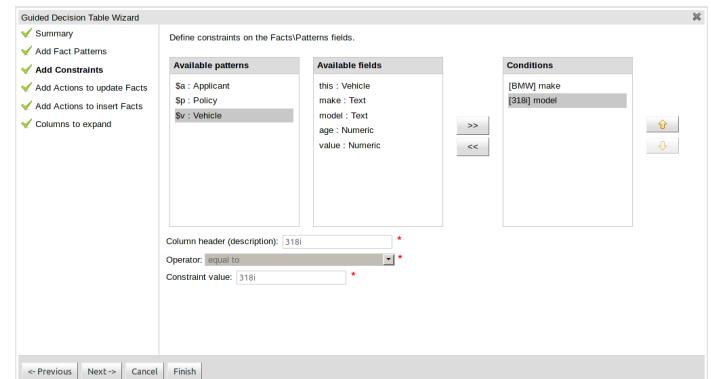
## Analysis

Find errors and omissions

min-age	max-age	policy type	make	model	premium	Analysis	
Applicant [Sa]					Policy [Sp]	Vehicle [Sv]	\$p
age [≥]	age [≤]	type [=]	make [=]	model [=]	premium		
18	25	TPFT	BMW	318i	1500		
18	10	COMP	BMW	318i	1500	Impossible match on age	
18	25	TPFT	BMW	M3	2000		
18	25	COMP	BMW	M3	1000		

## Let Guvnor guide you

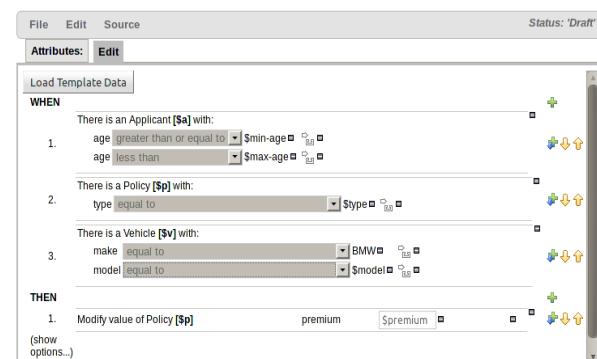
Wizard to construct different types  
 Avoid errors  
 Generate expanded form



# Guvnor - Templates

## Scaffolding for rules

Quickly build similar rules  
 Author structure with the guided rule editor  
 Insert place-holders for variables  
 Define values for variables



The screenshot shows the 'Edit' tab of a rule template in the Guvnor interface. The template structure is as follows:

```

WHEN
1. There is an Applicant [$a] with:
   age greater than or equal to $min-age
   age less than $max-age
2. There is a Policy [$p] with:
   type equal to $type
3. There is a Vehicle [$v] with:
   make equal to BMW
   model equal to $model

THEN
1. Modify value of Policy [$p]
   premium $premium

```

## Flexible authoring

Keyboard or mouse navigation  
 Keyboard or mouse multi-cell selection  
 Copy and Paste or rows  
 Sorting column data  
 Column resizing

Merging of cells  
 Rapid authoring and maintenance

Grouping of cells  
 Hide areas whilst editing others  
 Focus attention to sub-sections

Template Data					
	\$min-age	\$max-age	\$type	\$model	\$premium
[+]	18	25	TPFT	318i	1000
[+]	18	25	COMPREHENSIVE	318i	1500
[+]	18	25	TPFT	M3	2000
[+]	18	25	COMPREHENSIVE	M3	2500

Template Data					
	\$min-age	\$max-age	\$type	\$model	\$premium
[+]	18	25	TPFT	318i	1000
[+]			COMPREHENSIVE		1500
[+]			TPFT	M3	2000
[+]			COMPREHENSIVE		2500

Template Data					
	\$min-age	\$max-age	\$type	\$model	\$premium
[+]	18	25	TPFT	318i	1000
[+]			COMPREHENSIVE		1500
[+]	18	25	TPFT	M3	2000

# Guvnor – Decision Tables

## Flexible authoring

Keyboard or mouse navigation  
 Keyboard or mouse multi-cell selection  
 Copy and Paste or rows  
 Sorting column data  
 Column resizing

Merging of cells  
 Rapid authoring and maintenance

Grouping of cells  
 Hide areas whilst editing others  
 Focus attention to sub-sections

#	Description	min-age	max-age	policy type	make	model	premium
		Applicant [\$a]		Policy [\$p]	Vehicle [\$v]		\$p
		age [≥]	age [≤]	type [=]	make [=]	model [=]	premium
1	Applicant	18	25	TPFT	BMW	318i	1000
2				COMP			1500
3				TPFT		M3	2000
4				COMP			2500

#	Description	min-age	max-age	policy type	make	model	premium
		Applicant [\$a]		Policy [\$p]	Vehicle [\$v]		\$p
		age [≥]	age [≤]	type [=]	make [=]	model [=]	premium
1	Applicant	18	25	TPFT	BMW	318i	1000
2				COMP			1500
3		18	25	TPFT	BMW	M3	2000

## Integration

BRL columns  
 Ultimate in flexibility  
 Use BRL fragments in your decision table  
 Mix and match with regular column types

Use jBPM Work Items as actions  
 Call your favourite jBPM Work Items  
 Pass bound variables in Work Items  
 Use Result Parameters to update Facts

Condition column configuration (BRL fragment)

Column header (description): brlCondition

Hide column:

**WHEN**

There is an Applicant [\$applicant] with:

1. age greater than or equal to \$min  
age less than \$max

**Apply changes**

Column configuration (execute a Work Item)

Column header (description): Call echo service

Work Item Name: My Echo Service

Input Parameters: request \$request

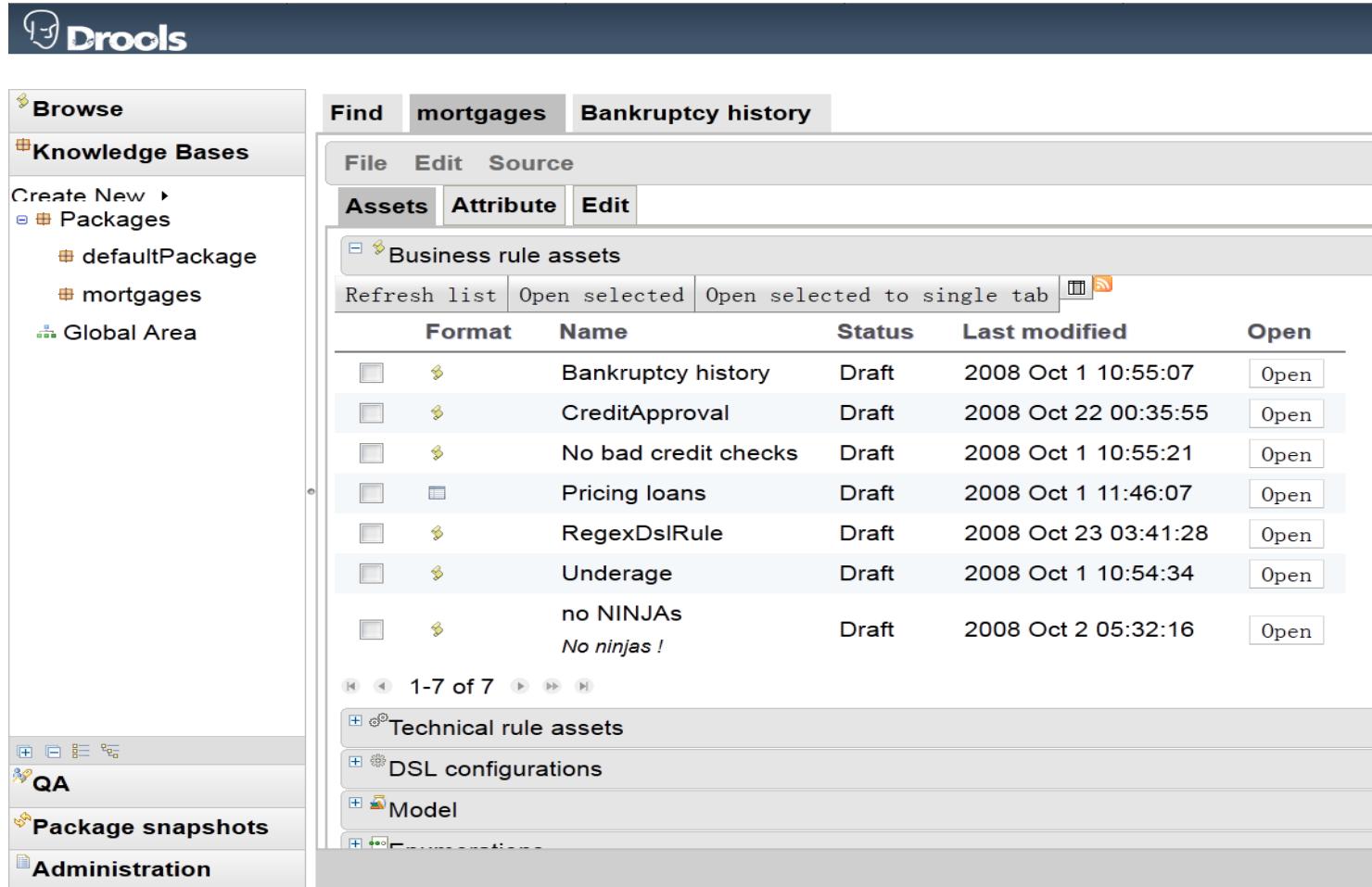
Hide column:

**Apply changes**



# Managing Assets With Guvnor

## Guvnor UI Overview:



The screenshot shows the Guvnor interface for managing business rule assets. The left sidebar includes links for Browse, Knowledge Bases (Create New, Packages, defaultPackage, mortgages, Global Area), QA, Package snapshots, and Administration. The main area has tabs for Find, mortgages, and Bankruptcy history. Under the Assets tab, the Business rule assets section is selected, displaying a list of assets with columns for Format, Name, Status, Last modified, and Open. The list includes:

Format	Name	Status	Last modified	Open
File	Bankruptcy history	Draft	2008 Oct 1 10:55:07	[Open]
File	CreditApproval	Draft	2008 Oct 22 00:35:55	[Open]
File	No bad credit checks	Draft	2008 Oct 1 10:55:21	[Open]
File	Pricing loans	Draft	2008 Oct 1 11:46:07	[Open]
File	RegexDslRule	Draft	2008 Oct 23 03:41:28	[Open]
File	Underage	Draft	2008 Oct 1 10:54:34	[Open]
File	no NINJAs No ninjas !	Draft	2008 Oct 2 05:32:16	[Open]

Below the asset list are sections for Technical rule assets, DSL configurations, and Model.

## Managing Assets:

### Asset Attribute View

Save/Archive/Copy/Rename/Move/Validate assets.

View and edit asset meta data

Asset life cycle management

Version management

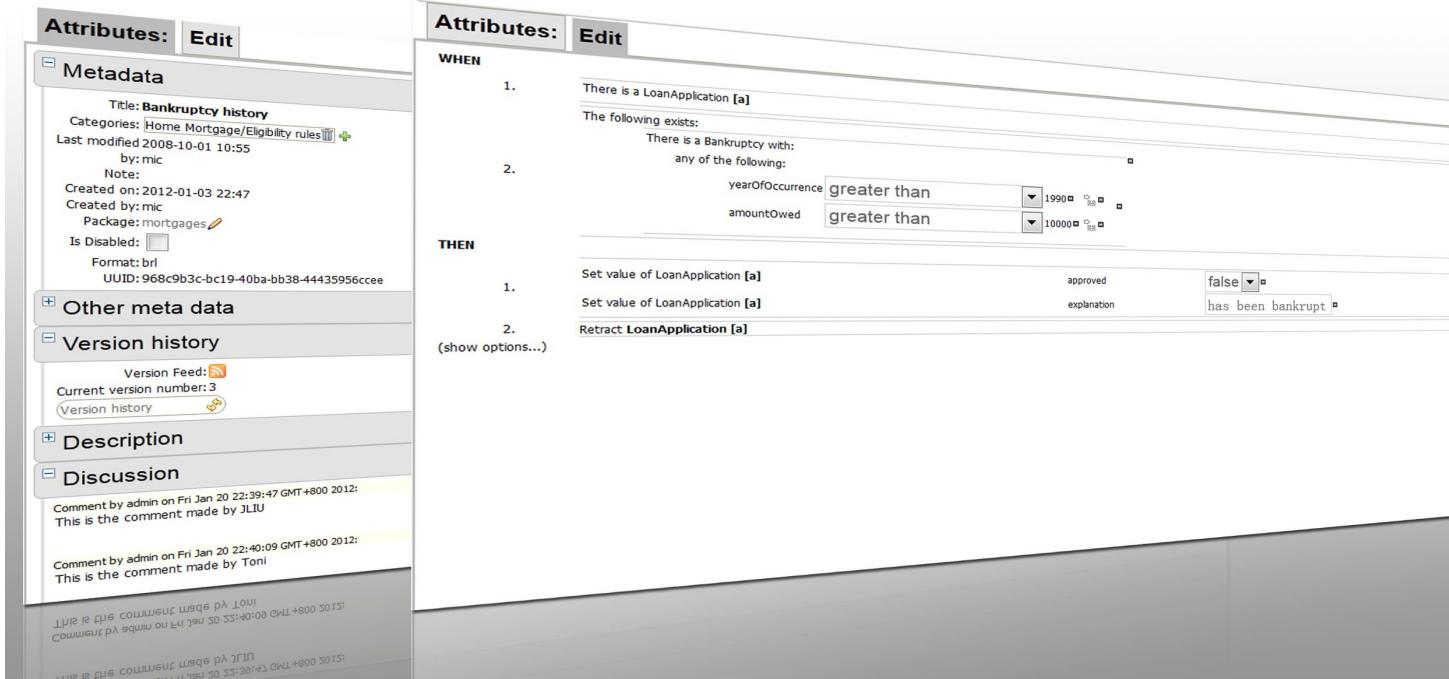
Discussion and discussion history

Atom/Feed

### Asset Edit View

Each asset type has its own editor to edit asset content.

# Managing Assets With Guvnor



The screenshot shows the 'Attributes' view for a 'Bankruptcy history' asset. The left sidebar contains sections for 'Metadata', 'Other meta data', 'Version history', 'Description', and 'Discussion'. The main panel displays the 'WHEN' and 'THEN' clauses of a rule:

```

WHEN
  1. There is a LoanApplication [a]
    The following exists:
      There is a Bankruptcy with:
        any of the following:
        yearOfOccurrence greater than 1990
        amountOwed greater than 10000
  2.

THEN
  1. Set value of LoanApplication [a] approved false
  2. Set value of LoanApplication [a] explanation has been bankrupt
  3. Retract LoanApplication [a]
  (show options...)

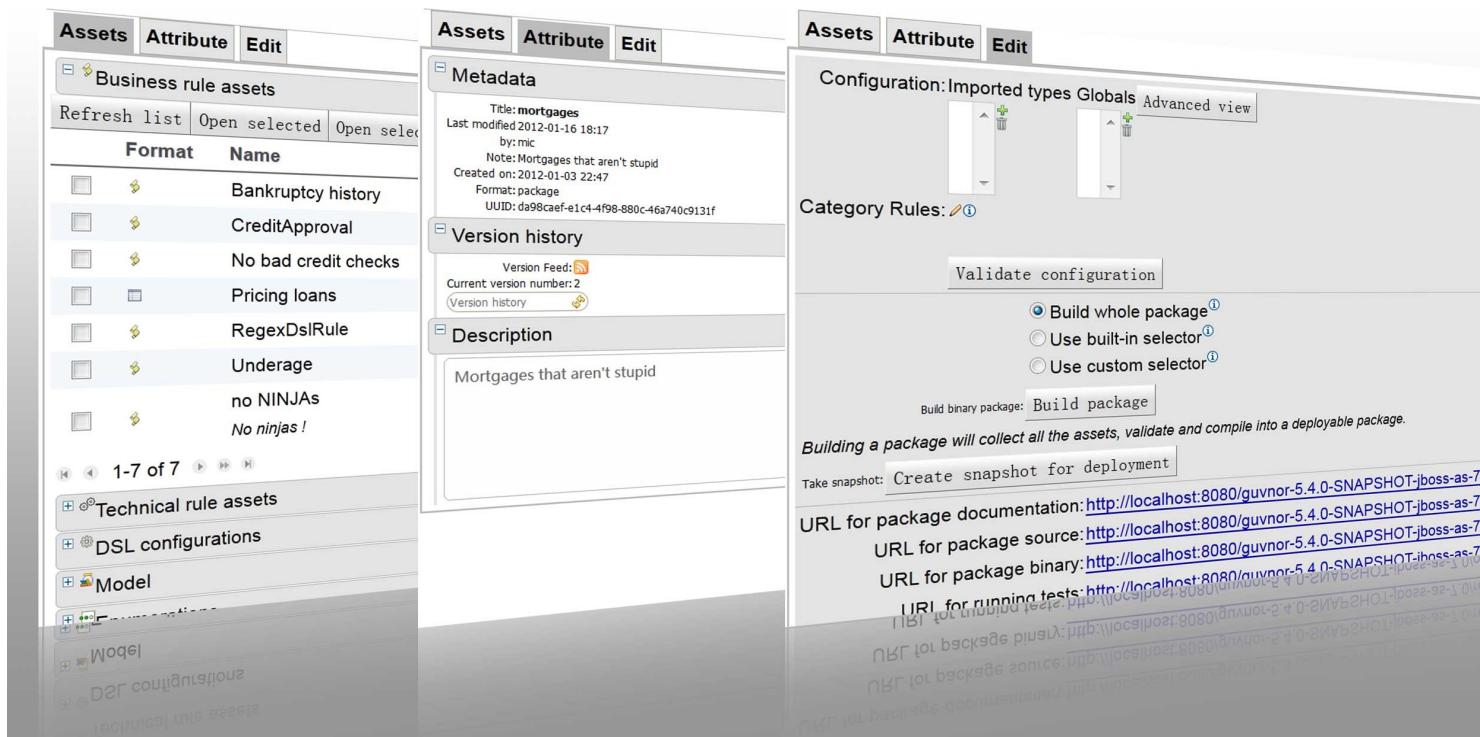
```

## Managing Packages:

Package assets list view: List all assets contained by the package based on asset types.

Package Attribute View – Same as Asset Attribute View

Package Edit View: Configure, build and deploy packages



The screenshot shows the 'Assets' view for a package named 'mortgages'. The left sidebar lists 'Business rule assets' (Bankruptcy history, CreditApproval, No bad credit checks, Pricing loans, RegexDslRule, Underage, no NINJAs, No ninjas!), 'Technical rule assets', 'DSL configurations', and 'Model'. The main panel shows the package's 'Metadata' (Title: mortgages, Last modified: 2012-01-16 18:17, by: mrc, Note: Mortgages that aren't stupid, Created on: 2012-01-03 22:47, Format: package, UUID: da98caeef-e1c4-4f98-880c-46a740c9131f), 'Version history' (Version Feed: RSS, Current version number: 2), and 'Description' (Mortgages that aren't stupid).

The right panel shows 'Configuration: Imported types Globals Advanced view' with a 'Category Rules' section and a 'Validate configuration' button. It also includes options for 'Build whole package' (selected), 'Use built-in selector', 'Use custom selector', and a 'Build binary package' button. Below this, there are links for 'Create snapshot for deployment', 'URL for package documentation', 'URL for package source', 'URL for package binary', and 'URL for running tests'.



# Guvnor for Drools and jBPM

## Guided Rule Editor

Traditional applications fall short on flexibility:

Disapprove loan when

applicant is younger than

applicant is older than  Invalid: not a number

With the **guided rule editor**, users can read, write and redefine business rules themselves:

**WHEN**

1. There is a Mortgage [m]

There is an Applicant with:

2. age greater than
- mortgage equal to

The following does not exist:

There is a Guarantor with:

3. age less than
- mortgage equal to

**THEN**

1. Set value of Mortgage [m] approved

With **DSL**, developers can build simple, readable rule pieces and users combine them as needed:

**WHEN**

1. There is a mortgage
2. Mortgage  has an applicant above
3. Mortgage  has no guarantor below

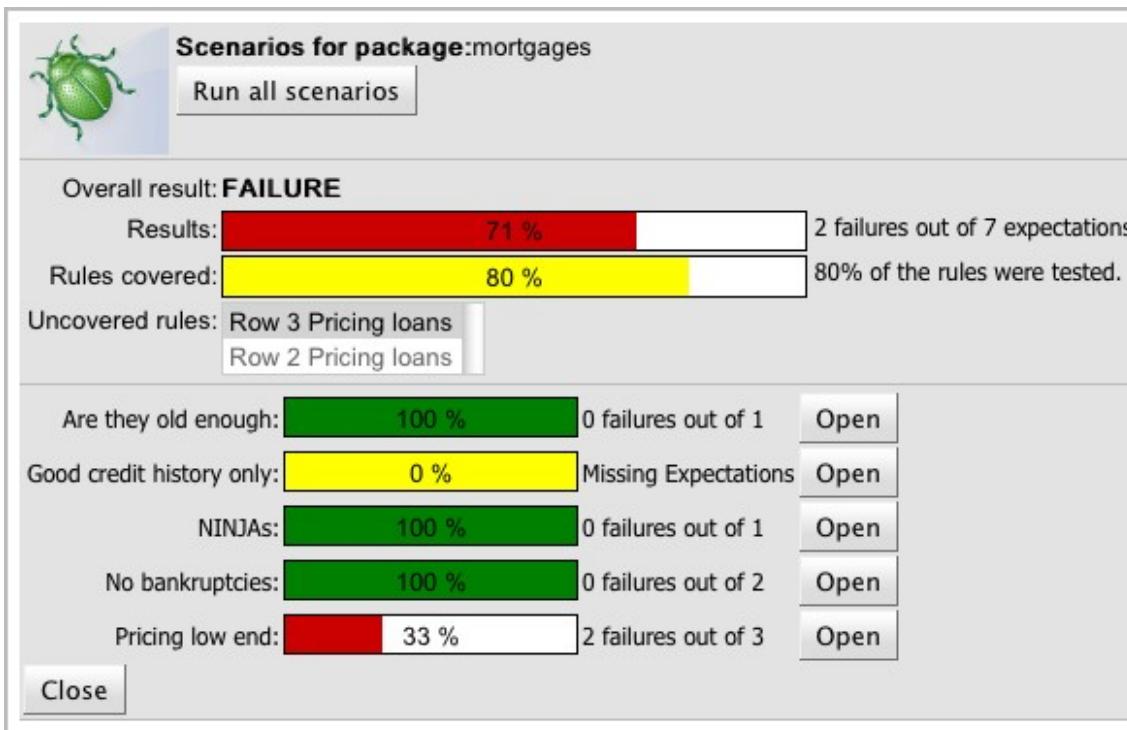
**THEN**

1. dissapprove mortage

## QA - Testing & Verification

### Scenario Tests

Test Scenarios are written by the rule authors to validate that the knowledge base is working.  
 Scenario report shows the test coverage and the test statuses.



### Package Analysis

Package analysis uses Drools rules to statically analyze knowledge modules.  
 It produces a report that contains information about the quality of the knowledge module.

- ✖ Errors (0 items).**
- ⚠ Warnings (11 items).**
  - ⊕ Rule 'RegexDslRule' has no RHS.
  - ⊕ Impacted rules:**
    - ⚡ RegexDslRule
  - ⊕ Rule 'Dummy rule' has no RHS.
  - ⊕ Rule base covers == Asset, but it is missing != Asset
  - ⊕ Rule base covers == 30, but it is missing != 30
- ☐ Notes (0 items).**

# Drools Planner

## *Planning optimization for Java™*

Every organization has planning problems, such as:

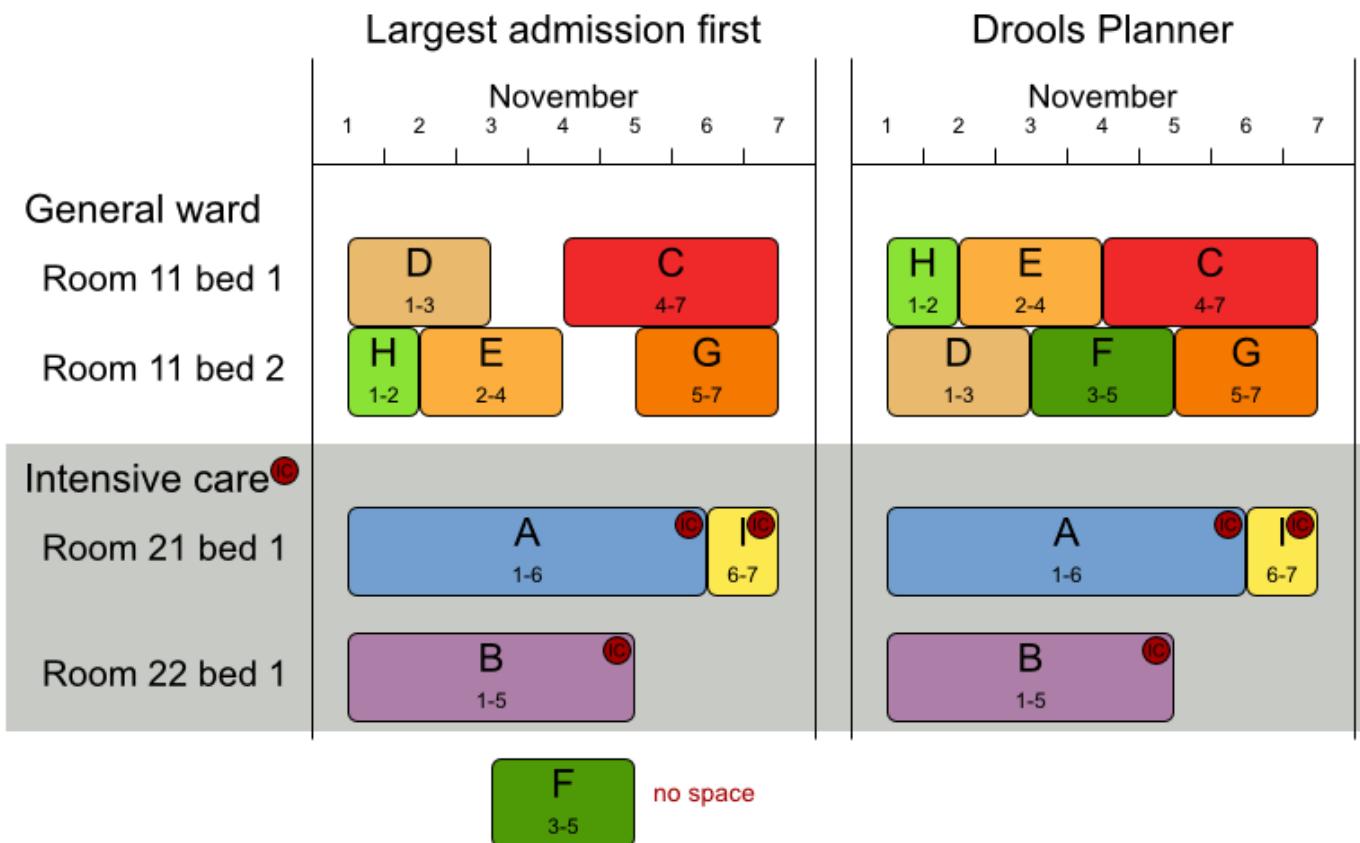
- Employee rostering
- Task scheduling
- Vehicle routing
- Assembly line optimization
- Bin packing

Yet, they hardly optimize those problems. Why? Because those problems are “NP-complete”: computationally very difficult and humanly impossible to optimize.

**A hospital can accommodate more patients when optimized with Drools Planner**

### Patient admission schedule

Assign each patient a hospital bed.



Better planning algorithms reduce costs, improve service quality and help save the environment:

Exam scheduling with Planner finds a feasible solution where common algorithms fail.

Nurse rostering with Planner has on average **53%** less unwanted shift assignments.

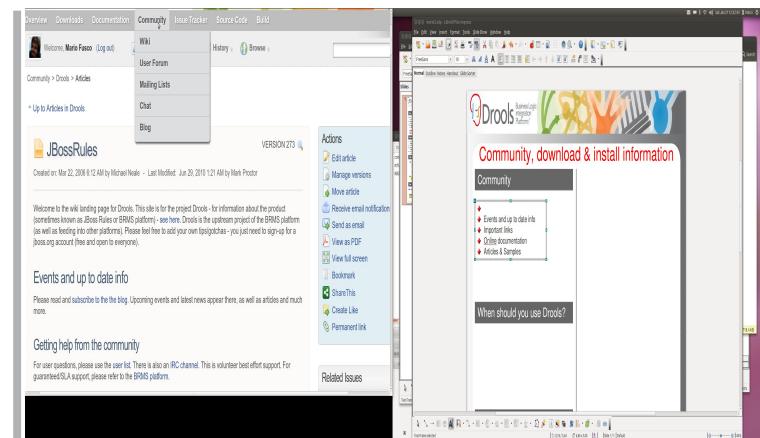
Process allocation to computers with Planner has on average **28%** less congested resources

To confirm these numbers for yourself, download Planner and run the examples.

# Community, download & install info

## Community Information

Events and up to date info  
 Important links  
 Online documentation  
 Articles & Samples  
 Tips, Tricks & FAQ for users (rule developers)  
 Patterns, best practices and guidelines  
 Presentations slides and videos

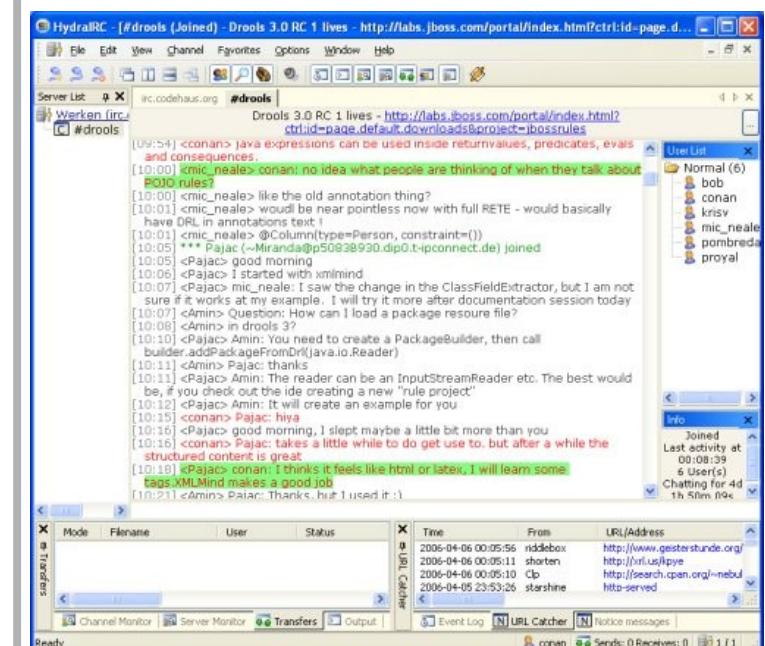


The screenshot shows the JBossRules article on the Drools community wiki. The page includes a sidebar with navigation links like 'Community', 'Issue Tracker', 'Source Code', and 'Build'. The main content area displays the article's text, which is a welcome message for visitors to the wiki.

<https://community.jboss.org/wiki/JBossRules>

## Getting help (and helping out)

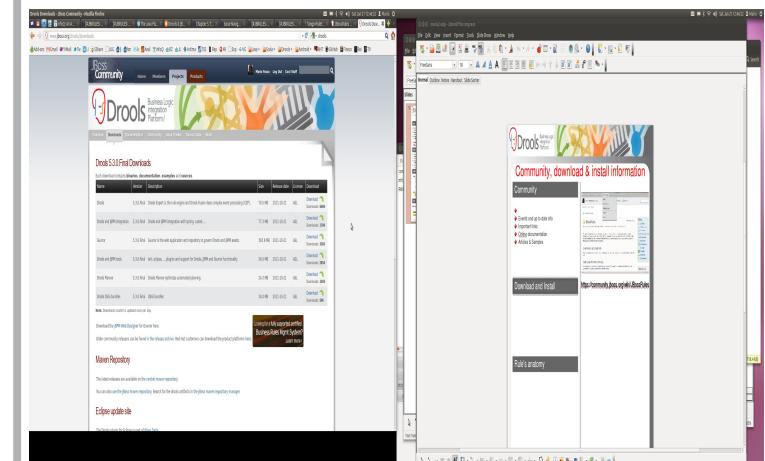
As normal, the maniacs can be found in the drools IRC room at irc.codehaus.org.  
 Web client available for those behind a firewall <http://irc.codehaus.org>  
 2 mailing list:  
 "user" list for end user issues  
 "dev" for questions about with the internals  
 Jira to check existing issues and log yours is available at  
<https://issues.jboss.org/browse/JBRULES>



The screenshot shows a HydraIRC window with multiple tabs. The '#drools' tab is active, displaying a log of a conversation between users like 'conan', 'mic\_neale', and 'Pajac'. The log shows various technical discussions related to Drools rule syntax and annotations.

## Download and Install

Download binaries, documentation, examples and sources in a single file  
 Projects are available on both the central Maven repository and on the JBoss one  
 Latest releases readily available including Beta and Candidate ones  
 Drools plugin for Eclipse available for download as part of the Jboss Tools at <http://www.jboss.org/tools/download.html> or using the standard Eclipse Marketplace Client



The screenshot shows two browser tabs side-by-side. The left tab is the 'Drools Download Overview' page, listing various binary distributions and source code options. The right tab is the 'JBoss Tools download' page, which includes a link to the Drools download section.

<https://www.jboss.org/drools/downloads>



# The importance of Product and Project

## Branding

JBoss BRMS is the branded product offering from Red Hat; for which you can buy support subscriptions and professional services, such as training and on-site consultancy. JBoss itself is a Red Hat brand for it's Java middleware.

JBoss BRMS is made up from a number of community projects - Drools Expert, Fusion, jBPM and Guvnor.

## Multi-Year Support

Red Hat commits to supporting a product release over a large number of years, the length varies from product to product. Cumulative patch releases ensure you get the fixes without the risk of new features. And the assurance that we will be there for the long term..org does not do maintenance releases it's a continuously forward moving R&D project. If you want the fixes you are going to get all the new features too, and the risks that come with them. Getting community support for previous .org releases can be hard, leaving you to the mercy of a volunteer based community.

## Multi-Year Support

Released products are tested across the range of JBoss and Red Hat products and also multiple versions. So if you are using JBoss BRMS you'll know it'll work on the range of AS services, or the JBoss SOA stack.

We also check compatibility across other platforms such as Websphere and even IBM z/OS. .org. Community releases do not go through this compatibility matrix level of testing.



Feature	Community	Enterprise
Open Source	x	x
Benefits from testing by worldwide Community	x	x
Recommended for Production Use		x
Patch Update & Service Pack Program		x
Security Errata Program		x
Automated Software Update & Alert Service		x
Defect & Feature Escalation & Prioritization Process		x
Developer Support		x
24x7 Production Support & Services		x
Platform Certifications & Training Certifications		x
Defined Support SLA and End-of-Life Policy		x
Out-of-the-Box Configured for Enterprise Use		x
Operations Management Tools		x
Platform testing & certification process		x
Redistribution of modified JBoss technologies		x
Red Hat Open Source Assurance (Legal Protection)		x

## Sanity through Sanitization

A common fear with OSS is due to the transparency of R&D. From casual inspection this can seem quite hairy as end users are exposed to all the unstable, unfinished and experimental works. This leaves a lot of uncertainty and fear in using OSS in a production environment.

Closed source companies get to do all their R&D behind closed doors, and end users are only exposed and aware of the highly polished marketing.

.com addresses this issue by removing or demarcating experimental or unstable features. To ensure there is a level of sanity and trust in what you are using. The product is typically 3 to 6 months behind in features, but offers strong levels of stability as a result.

## Unburden R&D

Everything that goes into ensuring a great product for the enterprise version entails a lot of work. You need a lot of resources that have meticulous attention to detail. Their priorities will be different to the R&D developers priorities. Stability and maintenance are the enemy of innovation. To ensure we have continued innovation at a project level it's important that we isolate R&D from these pressures. So everything about the product process is about freeing up the .org R&D developers so they can focus on ideas and innovations.

## Community Independence

By separating .org from .com it ensures that R&D can be bottom up user and community driven. This ensures a healthy eco system for ideas and innovation and collaboration - compared to a top down marketing driven model. The life cycle of .com and .org feeding into each other ensures the best of both worlds and help's maintain an important but delicate balance of innovation versus sanity and stability.

I should add that .org is still relevant and important for .com customers as it provides an open environment for them to get involved and upstream their work, which may eventually end up in the product.

## Direct Impact on Road Map

Customers you have a direct line to influence roadmaps and bugs. Resources are prioritized for customers versus community jira's or mailing lists posts. It is also the only way to get Red Hat engineering resources of legacy releases, they will ignore .org legacy issues in the community.

