

Drools & jBPM

All things Artificial Intelligence related: Rules, Processes, Events, Agents, Planning, Ontologies and more :)

FRIDAY, JULY 04, 2008

Drools 5.0 M1 - New and Noteworthy

Posted by Michael Neale

Drools 5.0 will be the launch of what we call the Business Logic integration Platform (BLiP) - the BRMs is dead :) The future is for a unified and integrated solution for Rules, Processes and CEP - this is what users want and this is what we are aiming for.

Drools 5.0 will split up into 4 main sub projects, the documentation has already been split to reflect this:

- Drools Guvnor (BRMS/BPMS)
- Drools Expert (rule engine),
- Drools Flow (process/workflow)
- Drools Fusion (cep/temporal reasoning)

M1 is still very hairy and only for the hard core drools users, little documentation has been updated - although some Flow and Guvnor stuff has been. So you will have to rely on looking at code and unit tests, as well as asking on the mailing lists and irc - we hope this new and noteworthy document helps guide you too.

M2 will involve an API change as we refactor away from being rules centric, as discussed [here](#), we will provide a legacy 4.0 wrapper jar for backwards compatability in some later milestones.

We hope that M3/M4 will start to be more user/public friendly as the feature set matures and bugs come in and we start to update the documentation. We are hoping for an August/Sept release.

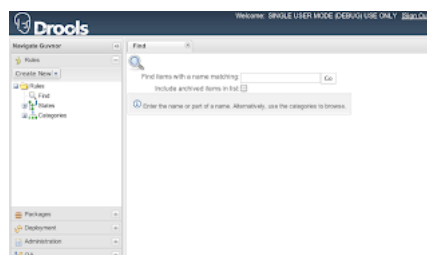
5.0 M1 can be found on the main [drools download](#) page. The Binary with dependencies is particularly large, but the bulk of it is uml javadocs, we will try and address this in M2 or M3 where we will try and remove non public stable apis.

A Big Thanks to the following Contributors

Ming Jin - Package serialisation performance increase (10x improvement)
Matthias Groch - sliding windows algorithm research
Tino Breddin - temporal operators
Matt Geis - new DSL parser and enhancements
Steven Williams - various decision table tasks.

GUVNOR (THE BRMS COMPONENT)

New look web tooling



Web based decision table editor

Decision table						
Modify...						
	Description	Advertiser type	age is at least	Postcode gre...	Postcode len...	Set the value...
1	Good suburbs	Agency	10	4000	4100	→ 42
2	Good suburbs	Agency	2000	2100	→ 43	Good region
4	Good suburbs	Agency	2200	2300	→ 43	Good region
Advertiser type: Partner (1 Row)						
3	Partners	Partner	1		→ 49	Other

Integrated scenario testing

DROOLS & JBPM JOB BOARD

[MORE JOBS >](#)
[POST A JOB >](#)

POWERED BY JOBTREAD



SUBSCRIBE TO

Posts

Comments

Subscribe email address:

Submit

[Archive Page](#)

DROOLS LINKS

[Overview](#)

[Download](#)

[Documentation](#)

[Free support](#)

[Paid support by Red Hat](#)

JBPM LINKS

[Overview](#)

[Download](#)

[Documentation](#)

[Free support](#)

[Paid support by Red Hat](#)

BLOG ARCHIVE

► [2020](#) (7)

► [2019](#) (6)

► [2018](#) (10)

► [2017](#) (24)

► [2016](#) (26)

► [2015](#) (30)

► [2014](#) (50)

► [2013](#) (60)

► [2012](#) (116)

► [2011](#) (152)

► [2010](#) (102)

► [2009](#) (146)

▼ [2008](#) (111)

► [December](#) (6)

► [November](#) (4)

► [October](#) (12)

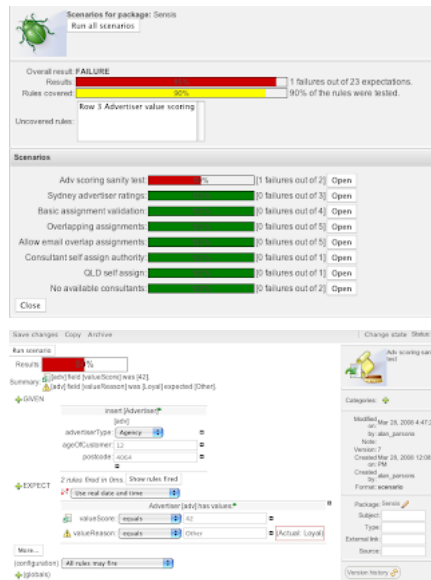
► [September](#) (8)

► [August](#) (15)

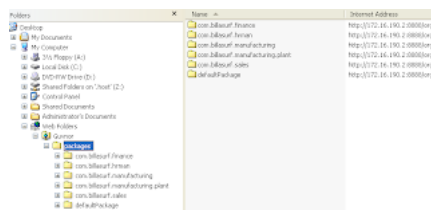
▼ [July](#) (15)

[New Drools Logos](#)

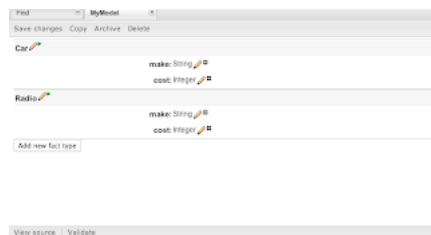
[Drools Flow view in Guvnor with GWT Diagram](#)



WebDAV file based interface to repository



Declarative modelling of types (types that are not in pojos)



This works with the new "declare" statement - you can now declare types in dri itself. You can then populate these without using a pojo (if you like). These types are then available in the rulebase.

Others:

- Logic verifier
- Improvements to guided editor (many)

CORE ENGINE

Asymmetrical Rete algorithm implementation

Shadow proxies are no longer needed. Shadow proxies protected the engine from information change on facts, which if occurred outside of the engine's control it could not be modified or retracted.

PackageBuilder can now build multiple namespaces

You no longer need to confine one PackageBuilder to one package namespace. Just keep adding your DRLs for any namespace and `getPackages()` returns an array of `Packages` for each of the used namespaces.

```
Package[] packages = pkgBuilder.getPackages();
```

RuleBase attachment to PackageBuilder

It is now possible to attach a RuleBase to a PackageBuilder, this means that rules are built and added to the rulebase at the same time. PackageBuilder uses the Package instances of the actual RuleBase as it's source, removing the need for additional Package creation and merging that happens in the existing approach.

```
RuleBase ruleBase = RuleBaseFactory.newRuleBase();
PackageBuilder pkgBuilder = new PackageBuilder( ruleBase, null );
```

Binary marshalling of stateful sessions

Stateful sessions can now be saved and resumed at a later date.

Pre-loaded data sessions can now be created.

Pluggable strategies can be used for user object persistence, i.e. hibernate or identity maps.

GuvnorNG

Rolodex Panel Assembly for Guvnor (Anton Arhipov)

Guvnor BRMS and Eclipse Synchronisation

Integrating Rolodex to Guvnor for Image Asset Types

SVN url changes & a funny video

Drools and WordNet

Drools Scalability

Building Decision Trees

Drools and Machine Learning

Drools Smooks Data Loader

The Rise of the Open Source BRMS

Drools 5.0 M1 - New and Noteworthy

Texas October Rules Fest and Drools Team Meeting

- ▶ June (8)
- ▶ May (5)
- ▶ April (6)
- ▶ March (11)
- ▶ February (12)
- ▶ January (9)

- ▶ 2007 (136)
- ▶ 2006 (19)

LABELS

#AI (3)
#AngularJS (1)
#bpmn (1)
#BPMS (2)
#BRMS (5)
#DashBuilder (4)
#dmn (3)
#dmn #bpmn #redhat #trisotech
#drools (1)
#DMN #DMCommunity #Drools (1)
#dmn #drools #trisotech (1)
#dmn #RHsummit #drools (1)
#Drools (21)
#drools #amadeus #RHsummit (1)
#drools #bpmnext2018 #blockchain (1)
#drools #dmn #dmncookbook
#brucesilver (1)
#drools #dmn #redhat #trisotech
#DMNQuickStart (1)
#drools #dmn #redhat #trisotech
#DMNQuickStart (1)
#drools #jbpm #dmn #bpmn #signavio
#redhat #RHsummit (1)
#drools #jbpm #github (1)
#drools #jbpm #optaplanner
#bootcamp #day (1)
#drools #jbpm #optaplanner
#droolsday (1)
#drools #jbpm #optaplanner #redhat
#summit2017 (1)
#drools #projectlead (1)
#drools #release (2)
#drools #se-radio (1)
#DroolsAI (6)
#DroolsRules (18)
#Eclipse (1)
#editor (1)
#Errai (3)
#FEEL (1)

Type Declaration

Drools now supports a new base construct called Type Declaration. This construct fulfils two purposes: the ability to declare fact metadata, and the ability to dynamically generate new fact types local to the rule engine. The Guvnor modelling tool uses this underneath.

One example of the construct is:

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

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

Declaring Fact Metadata

To declare and associate fact metadata, just use the @ symbol for each metadata ID you want to declare. Example:

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

Triggering Bean Generation

To activate the dynamic bean generation, just add fields and types to your type declaration:

```
declare Person
    name : String
    age : int
end
```

DSL improvements

A series of DSL improvements were implemented, including a completely new parser and the ability to declare matching masks for matching variables. For instance, one can constrain a phone number field to a 2-digit country code + 3-digit area code + 8-digit phone number, all connected by a "-" (dash), by declaring the DSL map like:

The phone number is {number:\d{2}-\d{3}-\d{8}}

Any valid java regexp may be used in the variable mask.

COMPLEX EVENT PROCESSING SUPPORT (TEMPORAL REASONING)

Drools 5.0 brings to the rules world the full power of events processing by supporting a number of CEP features as well as supporting events as first class citizens in the rules engine.

Event Semantics

Events are (from a rules engine perspective) a special type of fact that has a few special characteristics:

- they are immutable
- they have strong time-related relationships
- they may have clear lifecycle windows
- they may be transparently garbage collected after it's lifecycle window expires
- they may be time-constrained
- they may be included in sliding windows for reasoning

Event Declaration

Any fact type can assume an event role, and its corresponding event semantics, by simply declaring the metadata for it. Both existing and generated beans support event semantics:

```
# existing bean assuming an event role
import org.drools.test.StockTick
declare StockTick
    @role( event )
end

# generated bean assuming an event role
declare Alarm
    @role( event )
    type : String
    timestamp : long
end
```

Entry-Point Stream Listeners

A new key "from entry-point" has been added to allow a pattern in a rule to listen on a stream, which avoids the overhead of having to insert the object into the working memory where it is potentially reasoned over by all rules.

```
$st : StockTick( company == "ACME", price > 10 ) from entry-point "stock stream"
```

#fiddle (1)

#FP (1)

#graalvm (1)

#GWT (10)

#javascript (1)

#jBPM (23)

#js (1)

#Kie (1)

#kogito (1)

#opensearch #drools (1)

#OptaPlanner (5)

#packt #books (3)

#polyglot (1)

#quarkus (1)

#recursion (1)

#redhat (1)

#UberFire (7)

#visualization (1)

#YCombinator (1)

2014 (1)

6.2.0.Final (1)

accumulate (2)

accumulate function (2)

actors (1)

actuators (1)

Ad-Hoc (1)

AI (8)

aires (1)

algorithm (2)

Analytics (2)

android (1)

announcement (1)

ANTLR (5)

API (1)

argentina (1)

arm (1)

Atom (1)

Backward Chaining. (5)

BAM (1)

Barcelona (5)

Bayesian (1)

Belief Systems (1)

benchmark (1)

BOF (4)

book (4)

Boot Camp (15)

bootcamp (3)

BPEL (1)

bpm (8)

BPMN (13)

bpmn2 (6)

bpms (3)

bpsim (1)

Brazilian (1)

BRMS (37)

brms ajax web (2)

BRMS Guvnor Drools (16)

brms insurance demo standalone (1)

buenos (1)

builder (1)

business (2)

business process (3)

Business Processes (4)

Business Rules (18)

Business Rules Forum (3)

Business Rules Governance (6)

To insert facts into an entry point:

```
WorkingMemoryEntryPoint entry = wm.getWorkingMemoryEntryPoint( "stock stream" );
entry.insert( ticker );
```

[StreamTest](#) shows a unit for this.

Event Correlation and New Operators

Event correlation and time based constraint support are requirements of event processing, and are completely supported by Drools 5.0. The new, out of the box, time constraint operators can be seen in these test case rules: [test_CEP_TimeRelationalOperators.drl](#)

As seen in the test above, Drools supports both: primitive events, that are point in time occurrences with no duration, and compound events, that are events with distinct start and end timestamps.

The complete list of operators are:

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

Sliding Time Windows

Drools 5.0 adds support for reasoning over sliding windows of events. For instance:

```
StockTick( symbol == "RHAT" ) over window:time( 60 )
```

The above example will only pattern match the RHAT stock ticks that happened in the last 60 clock ticks, discarding any event older than that.

Session Clock

Enabling full event processing capabilities requires the ability to configure and interact with a session clock. Drools adds support for time reasoning and session clock configuration, allowing it to not only run real time event processing but also simulations, what-if scenarios and post-processing audit by replaying a scenario.

The Clock is specified as part of the SessionConfiguration, a new class that is optionally specified at session creation time:

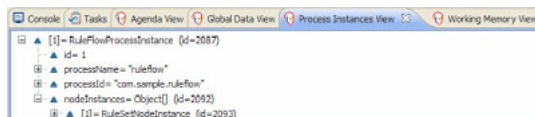
```
SessionConfiguration conf = new SessionConfiguration();
conf.setClockType( ClockType.PSEUDO_CLOCK );
StatefulSession session = ruleBase.newStatefulSession( conf );
```

DROOLS FLOW

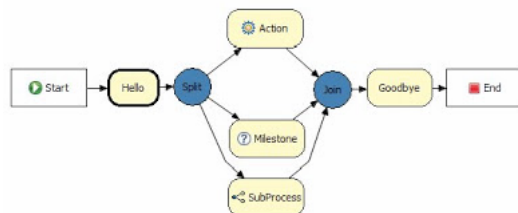
Drools 4.0 had simple "RuleFlow" which was for orchestrating rules. Drools 5.0 introduces a powerful (extensible) workflow engine. It allows users to specify their business logic using both rules and processes (where powerful interaction between processes and rules is possible) and offers a unified environment.

Interactive Debugger

Process Instance view at a specific breakpoint:



Current active nodes in a workflow in a specific breakpoint:



New Nodes

Timers:

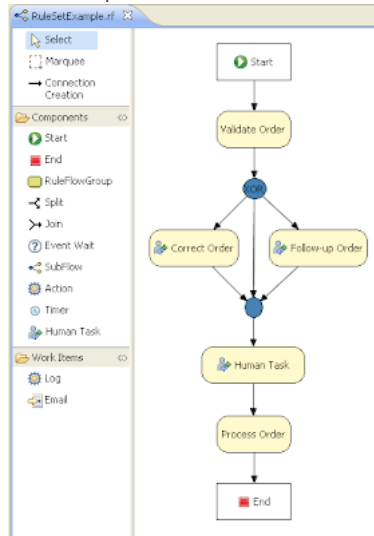
A timer node can be added which causes the execution of the node to wait for a specific period. Currently just uses JDK defaults of initial delay and repeat delay, more complex timers will be available in further milestones.

Camel	(4)
case management	(1)
CEP	(15)
Clips	(4)
cloud	(3)
Codehaus	(1)
combinatorial optimization	(1)
community	(5)
competition	(3)
compilers	(4)
Computer Games	(9)
computer vision	(1)
conference	(2)
configuration	(1)
Conflict Resolution	(2)
console	(1)
constraint programming	(2)
Content Management System	(1)
contribution	(1)
cookbook	(1)
Debug	(1)
decision	(1)
decision management	(1)
decision model and notation	(1)
decision tables	(15)
DecisionCamp	(6)
declarative programming	(2)
demo	(1)
designer	(2)
developer	(1)
developer's guide	(1)
development	(1)
DevNation	(3)
devvxx	(2)
DMCommunity	(1)
dmn	(1)
docker	(2)
document	(1)
Domain Specific Languages	(5)
DotNet	(2)
DRL	(8)
Drools	(484)
Drools Boot Camp	(29)
Drools Chance	(6)
Drools Expert	(27)
Drools Flow	(31)
Drools Fusion	(11)
drools puzzle	(4)
drools webinar	(1)
DroolsAI	(2)
Droos	(1)
droos ide update-site downloads	(1)
DSL	(1)
DSL regexp antlr	(1)
DynaBeans	(1)
dynamically generated classes	(1)
Eclipse	(8)
electronics	(1)
embedded	(1)
engine	(2)
ESP	(2)
evaluation process	(1)
event	(13)
examination	(2)
example	(2)
execution server	(1)

Human Task:

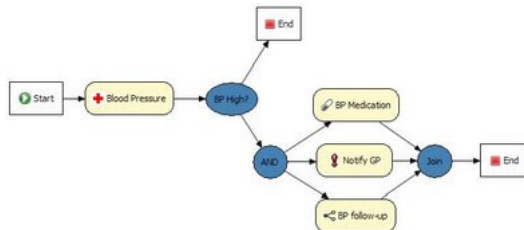
Processes can include tasks that need to be executed by human actors. Human tasks include parameters like taskname, priority, description, actorId, etc. The process engine can easily be integrated with existing human task component (like for example a WS-HumanTask implementation) using our pluggable work items (see below). Swimlanes and assignment rules are also supported.

The palette in the screenshot shows the two new components, and the workflow itself shows the human task in use. It also shows two "work items" which is explained in the next section:

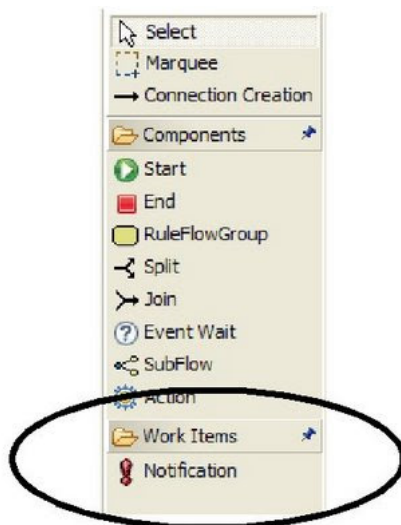
**Domain Specific Work Items**

Domain Specific Work Items are pluggable nodes that users create to facilitate custom task execution. They provide an api to specify a new icon in the palette and gui editor for the tasks properties, if no editor gui is supplied then it defaults to a text based key value pair form. The api then allows execution behaviour for these work items to be specified. By default the Email and Log work items are provided. The Drools flow Manual has been updated on how to implement these.

The below image shows three different work items in use in a workflow, "Blood Pressure", "BP Medication", "Notify GP":



This one owns a new "Notificatoin" work item:

**extensible Process Definition Language (ePDL)**

Drools 4.0 used Xstream to store its content, which was not easily human writeable. Drools 5.0 introduced the ePDL which is a XML specific to our process language, it also allows for domain specific extensions which has been talked about in detail in this blog posting "[Drools Extensible Process Definition Language \(ePDL\) and the Semantic Module Framework \(SMF\)](#)". An example of the XML language, with a DSL extension in red, is shown below.

Expert Systems (9)
 expressiveness (3)
 extensibility (1)
 FactTemplate (1)
 FedEx (1)
 flamegraph (1)
 form (1)
 Form Builder (3)
 forms (1)
 Forward Chaining (1)
 fraction (1)
 free (2)
 functional programming (1)
 Fuzzy (1)
 generated classes (1)
 GIS (2)
 Google Summer of Code (1)
 grammar (1)
 GSoC (10)
 gsoc2012 (1)
 GUI (9)
 Guice (1)
 guide (1)
 Guvnor (50)
 gwt (4)
 gwt-maven-plugin (1)
 hardware (1)
 heal (1)
 Healthcare (9)
<http://www.blogger.com/img/blank.gif> (1)
 human (1)
 IKVM (1)
 image processing (1)
 improvements (1)
 infinispn (1)
 infoQ (1)
 integration (3)
 IntelliFest (2)
 interface (1)
 interview (1)
 ireland (1)
 Janino (1)
 java (6)
 JavaOne (3)
 javapolis (4)
 JBoss (1)
 JBoss Rules (54)
 jBPM (124)
 jbpn-console-ng (4)
 jbpn. 6.2.0.Final (1)
 jBPM5 (29)
 jBPM5 webinar (3)
 jbpn6 (6)
 JDT (1)
 Jess (2)
 JFDI (1)
 Job (11)
 JSON (1)
 JUG (5)
 junit (1)
 KAMS (1)
 kbase (1)
 kie (8)
 kogito (1)
 runtime (1)

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

<nodes>
  <start id="0" />

  <action id="1" dialect="java">
    list.add( "action node was here" );
  </action>

  <mysdl:logger id="2" type="warn">
    This is my message
  </mysdl:logger>

  <end id="3" />
</nodes>

<connections>
  <connection from="0 to="1" />
  <connection from="1" to="2" />
  <connection from="2" to="3" />
</connections>

</process>
```

Pluggable Nodes

The underlying nodes for the framework are completely pluggable making it simple to extend and to implement other execution models. We already have a partial implementation for [OSWorkflow](#) and are working with [Deigo](#) to complete this to provide a migration path for OSWorkflow users.

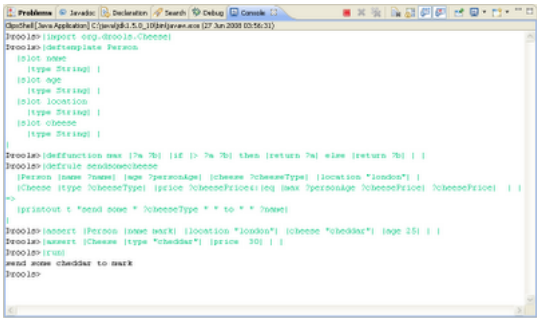
Other enhancements include exception scopes, the ability to include on-entry and on-exit actions on various node types, integration with our binary persistence mechanism to persist the state of long running processes, etc. Check out the [Drools Flow](#) documentation to learn more.

DROOLS CLIPS

A very alpha quality version of Drools Clips is now working and supports:

- deftemplate
- defrule
- deffunction
- and/or/not/exists/test Conditional Elements
- Literal, Variable, Return Value and Predicate field constraints

You can look at the [ClipsShellTest](#) and [LhsClipsParserTest](#) get an idea of the full support. It's still early stages and it's very rough in places, especially on error handling and feedback as well as no view commands to display data. The Shell in action:



The screen shot is a contrived example but it does show a shell environment cleanly mixing deftemplates and pojoes - note that Drools 5.0 does not require shadow facts, due to the new asymmetrical Rete algorithm. It also shows deffunction in use.

Share / Save



at 1:23 pm 

Labels: [Drools](#), [release](#)

8 comments:

 [woolfel](#) Friday, July 04, 2008

Congrats on the progress. Have you guys done any performance or stress testing of the timer based approach? I'm curious to see how well it scales with respect to interval density, large set of facts and multiple data streams.

ksession	(1)
kubernetes	(1)
Latam	(1)
LDAP	(1)
Life Cycle	(1)
lists	(1)
Logic Operators	(3)
london	(4)
machine learning	(3)
management	(1)
maven	(1)
medical	(2)
meetup	(1)
Meetups	(5)
mgwt	(1)
micro service	(1)
MicroContainer	(1)
microservices	(2)
milestone	(1)
Mind Map	(1)
MISMO	(2)
mobile	(1)
modify block	(1)
Monitoring	(1)
motors	(1)
MVEL	(8)
MySQL	(1)
Natural Language	(4)
Negation	(1)
October Rules Fest	(14)
one	(1)
Open Source	(4)
openshift	(1)
ORF	(8)
ORF 2008	(1)
OSGi	(1)
parser	(2)
patterns	(1)
Performance	(19)
persistence	(2)
PHREAK	(1)
planner	(87)
plug tree	(1)
PMML	(1)
Portuguese	(1)
Presentation	(44)
Probability	(1)
process	(7)
processes	(2)
Production Rules Systems	(2)
Programming	(4)
Progress	(4)
project	(1)
project proposals	(2)
quote	(1)
Red Hat	(1)
Red Hat Summit	(3)
relational programming	(1)
release	(27)
remote	(1)
repository	(1)
Research Network	(2)
REST	(1)
Rete	(18)
RIF	(2)
robot	(1)

Some general thoughts on the topic of timers. If the timers are defined by the rules and sliding windows, performance starts to become a serious performance issue with complex rulesets that perform joins. For simple examples like the ones Espers uses with 1 object type, it's probably ok.

From the research Karl did at Aachen, when the interval is shorter than 1 second, the timer ends up eating all the CPU cycles. that's a hard limit regardless of the rule engine. Even in the case where the intervals are greater than 10 seconds, the timer could still eat all CPU cycles with large complex rulesets and large datasets greater than 250K. In this case, the kind of timer I'm thinking of uses 1 timer per rule. Which means 500 rules with time windows would mean the rule engine has 500 timers. If we have a ruleset of 2K rules, that would mean 2K timers.

I favor a lazy approach, which adds minimal overhead. when I profiled the lazy implementation in jamocha, the overhead was less than 1%. In my temporal logic paper, I propose a different kind of timer, which is controlled by the rule engine and does periodic house cleaning at the engine level, not at the rule level.

It would be good to document the design, so users can use it properly.

[Reply](#)



Mark Proctor Friday, July 04, 2008

We don't use any standard Timers. What we do is calculate the delay for the next execution and sleep till then, when it executes it pops the next one off the stack, calculates the delay and sleeps again until it needs to fire. A java.util.timer per rule would indeed be far too heavy.

We haven't done any performance work yet, we know there is a lot to do, especially for multiple thread usage.

[Reply](#)



woolfel Friday, July 04, 2008

sounds like drools 5 uses a pool of timers. does that mean it is still 1 timer per rule?

Even if the rule engine doesn't use standard java timer, having 1 timer per rule like espers isn't going to scale well for hundreds or thousands of rules. I had a discussion with ernest about the use of timers in a rule engine and he had similar concerns.

[Reply](#)

Anonymous Friday, July 04, 2008

Ahh (penny drops) ... a lot of the features you were talking about make a lot more sense now.

They Guvnor looks good. Been playing wiht the latest versions for a while now

Quick question about the Declarative modelling. Where's the best place (in the code?)to find out more info?

Paul

[Reply](#)



Mark Proctor Friday, July 04, 2008

heh, glad you are finally getting it. Will be easier to discuss the details on IRC, see you there.

Mark

[Reply](#)



woolfel Friday, July 04, 2008

I don't know if you saw the post I started on the CEP forum, but one thing that feels wrong to me with CEP is the notion that all events are persistent and immutable. From the CEP section of the blog entry, I see drools is following the "immutable" definition.

My gut tells me that approach will prove to be wrong in the long term. Clearly, not all events should be persistent, nor should all events be immutable. Just because some events are immutable, doesn't mean all events really are immutable :)

I think Tim Bass is dead on, when he says current CEP products are in the hype cycle.

[Reply](#)



Mark Proctor Friday, July 04, 2008

I agree, we've taken this approach though as it helps simplify things and we can solve a large number of use cases with this approach. When we find valid use cases for mutable events we will consider adding support for that.

[Reply](#)



Geoffrey De Smet Sunday, July 06, 2008

Another new and noteworthy is that it contains the first alfa release of drools-solver.

If you're interested in planning problems, take a minute to download the zip and take a look at the examples.

[Reply](#)

[robotics](#) (1)

[rollback](#) (1)

[rule](#) (2)

[Rule Authoring](#) (6)

[Rule Engines](#) (22)

[Rule Flow](#) (14)

[RuleML](#) (8)

[RuleML 2008](#) (1)

[Rules](#) (14)

[Rules Engine](#) (6)

[runtime](#) (1)

[salaboy](#) (1)

[Savvion](#) (5)

[SBVR](#) (2)

[Seam](#) (3)

[search space](#) (3)

[semantic web](#) (3)

[sensors](#) (1)

[sequential](#) (2)

[server](#) (1)

[serverless](#) (1)

[Service Manager](#) (1)

[service repository](#) (1)

[shadow facts](#) (2)

[shadow proxies](#) (1)

[simulation](#) (1)

[Simulation and Testing](#) (2)

[Smooks](#) (2)

[SOA](#) (4)

[solver](#) (25)

[source code](#) (1)

[south america](#) (1)

[Spring](#) (6)

[SSL](#) (1)

[standards](#) (3)

[Stream](#) (1)

[student](#) (1)

[summit](#) (1)

[superdevmode](#) (1)

[swarm](#) (1)

[SwitchYard](#) (1)

[Synasc](#) (1)

[syntax](#) (2)

[task](#) (1)

[tasks](#) (2)

[Templates](#) (2)

[Testing](#) (1)

[tests](#) (1)

[Time](#) (1)

[Time-Sensitive](#) (1)

[Tomcat](#) (1)

[Training](#) (2)

[traits](#) (2)

[traveling tournament](#) (1)

[UberFire](#) (8)

[uncertainty](#) (1)

[units](#) (1)

[upgrade tool](#) (1)

[use case](#) (3)

[user](#) (2)

[user interface](#) (1)

[variables](#) (1)


[video](#) (9)

[videos](#) (1)

[Web Services](#) (5)

[webinar](#) (3)

Enter your comment...

 Comment as: PaulSpencer (C ▾)

Sign out

Publish

Preview

☐ Notify me

[Newer Post](#)

[Home](#)

[Older Post](#)

Subscribe to: [Post Comments \(Atom\)](#)

- [wildfly \(1\)](#)
- [Windows \(1\)](#)
- [WordNet \(1\)](#)
- [workbench \(6\)](#)
- [workflow \(1\)](#)
- [workshop \(14\)](#)
- [XAI \(1\)](#)

CONTRIBUTORS

- [Andrew Waterman](#)
- [Bob Brodt](#)
- [Eder Ignatowicz](#)
- [Edoardo Vacchi](#)
- [Edson Tirelli](#)
- [Gabriele Cardosi](#)
- [Geoffrey De Smet](#)
- [Guilherme Carreiro](#)
- [Jaroslaw Kijanowski](#)
- [Joe White](#)
- [John Graham](#)
- [Kris Verlaenen](#)
- [LeoGomes](#)
- [Maciej Swiderski](#)
- [Marian Buenosayres](#)
- [Mario Fusco](#)
- [Mark Proctor](#)
- [Matteo Mortari](#)
- [Michael Neale](#)
- [Sotty](#)
- [Tiago Bento](#)
- [Tihomir Surdilovic](#)
- [Toni Rikkola](#)
- [Unknown](#)
- [Unknown](#)
- [Unknown](#)
- [Unknown](#)
- [Unknown](#)
- [Unknown](#)
- [Unknown](#)
- [asd](#)
- [manaswinidas](#)
- [paul browne](#)
- [porcelli](#)
- [salaboy](#)