



Criado por Rafael Luiz dos Santos em LibreOffice Impress como componente curricular da disciplina de IA (Inteligência Artificial) da Ufal – Universidade Federal de Alagoas – Campus A. C. Simões.

O que é Drools?

Uma ferramenta para gerência de regras de negócio permitindo melhor gerenciamento dos papéis dos profissionais no projeto.

Aplicações em IA na perspectiva de criar regras de inferência e na área de ML (Machine Learning).

Prós e contras

- Vantagens:
 - Regras rodam em uma camada separada do código;
 - Regras são mais naturais a um ser humano;
 - Facilidade de manutenção e leitura;
 - Não é preciso compilar um programa novamente.
 - Permite declarar a regra de negócio de forma mais declarativa;
 - Um próprio analista de negócios pode realizar a autoria e manutenção das regras.
- Desvantagens:
 - Requer uma curva de aprendizado (mesmo sendo pequena);
 - Em alguns casos um desenvolvedor deve tratar uma regra ambígua ou mal escrita.

Prerequisites para usabilidade

- JDK 11+ with JAVA_HOME configured appropriately
- Apache Maven 3.8.6+
- Optionally, an IDE, such as IntelliJ IDEA, VSCode or Eclipse

Criação de projeto

- **Comando:** “*mvn archetype:generate -DarchetypeGroupId=org.kie -DarchetypeArtifactId=kie-drools-exec-model-ruleunit-archetype -DarchetypeVersion=8.44.0.Final*”
- Depois preencher de acordo com a imagem abaixo:

```
Define value for property 'groupId': org.example
Define value for property 'artifactId': my-project
Define value for property 'version' 1.0-SNAPSHOT: :
Define value for property 'package' org.example: :
...
Y: : Y
...
[INFO] BUILD SUCCESS
```

Estrutura do projeto



Conhecendo o projeto Maven Apache

```
seminario > src > main > java > org > example > J Measurement.java > ...  
2  package org.example;  
3  
4  public class Measurement {  
5      private String id;  
6      private String val;  
7  
8      public Measurement(String id, String val) {  
9          super();  
10         this.id = id;  
11         this.val = val;  
12     }  
13  
14     public String getId() {  
15         return id;  
16     }  
17  
18     public String getVal() {  
19         return val;  
20     }  
21  
22     @Override  
23     public String toString() {  
24         StringBuilder builder = new StringBuilder();  
25         builder.append("Measurement [");  
26         if (id != null)  
27             builder.append("id=").append(id).append(", ");  
28         if (val != null)  
29             builder.append("val=").append(val);  
30         builder.append("]");  
31         return builder.toString();  
32     }  
33 }
```

Conhecendo o projeto Maven Apache

```
seminario > src > main > java > org > example > J MeasurementUnit.java > ...  
1  
2 package org.example;  
3  
4 import java.util.HashSet;  
5 import java.util.Set;  
6  
7 import org.drools.ruleunits.api.DataSource;  
8 import org.drools.ruleunits.api.DataStore;  
9 import org.drools.ruleunits.api.RuleUnitData;  
10  
11 public class MeasurementUnit implements RuleUnitData {  
12  
13     private final DataStore<Measurement> measurements;  
14     private final Set<String> controlSet = new HashSet<>();  
15  
16     public MeasurementUnit() {  
17         this(DataSource.createStore());  
18     }  
19  
20     public MeasurementUnit(DataStore<Measurement> measurements) {  
21         this.measurements = measurements;  
22     }  
23  
24     public DataStore<Measurement> getMeasurements() {  
25         return measurements;  
26     }  
27  
28     public Set<String> getControlSet() {  
29         return controlSet;  
30     }  
31 }  
32
```


Conhecendo o projeto Maven Apache

```
seminario > src > test > java > org > example > RuleTest.java > RuleTest > test()

18 static final Logger LOG = LoggerFactory.getLogger(clazz:RuleTest.class);
19
20 @Test
21 public void test() {
22     LOG.info(msg:"Creating RuleUnit");
23     MeasurementUnit measurementUnit = new MeasurementUnit();
24
25     RuleUnitInstance<MeasurementUnit> instance = RuleUnitProvider.get().createRuleUnitInstance
26     try {
27         LOG.info(msg:"Insert data");
28         measurementUnit.getMeasurements().add(new Measurement(id:"color", val:"red"));
29         measurementUnit.getMeasurements().add(new Measurement(id:"color", val:"green"));
30         measurementUnit.getMeasurements().add(new Measurement(id:"color", val:"blue"));
31         measurementUnit.getMeasurements().add(new Measurement(id:"color", val:"yellow"));
32         measurementUnit.getMeasurements().add(new Measurement(id:"color", val:"brown"));
33
34         LOG.info(msg:"Run query. Rules are also fired");
35         System.out.println("TUDO OK!!\n\n");
36         List<Measurement> queryResult = instance.executeQuery(query:"FindColor").toList(identi
37
38         assertEquals(expected:5, queryResult.size());
39         assertTrue(message:"contains red", measurementUnit.getControlSet().contains("red"));
40         assertTrue(message:"contains green", measurementUnit.getControlSet().contains("green"));
41         assertTrue(message:"contains blue", measurementUnit.getControlSet().contains("blue"));
42         assertTrue(message:"contains yellow", measurementUnit.getControlSet().contains("yellow"));
43         assertTrue(message:"contains brown", measurementUnit.getControlSet().contains("brown"));
44     } finally {
45         instance.close();
46     }
47 }
48 }
```

Antes de conhecer as regras...

WHEN

<conditions>

THEN

<actions>

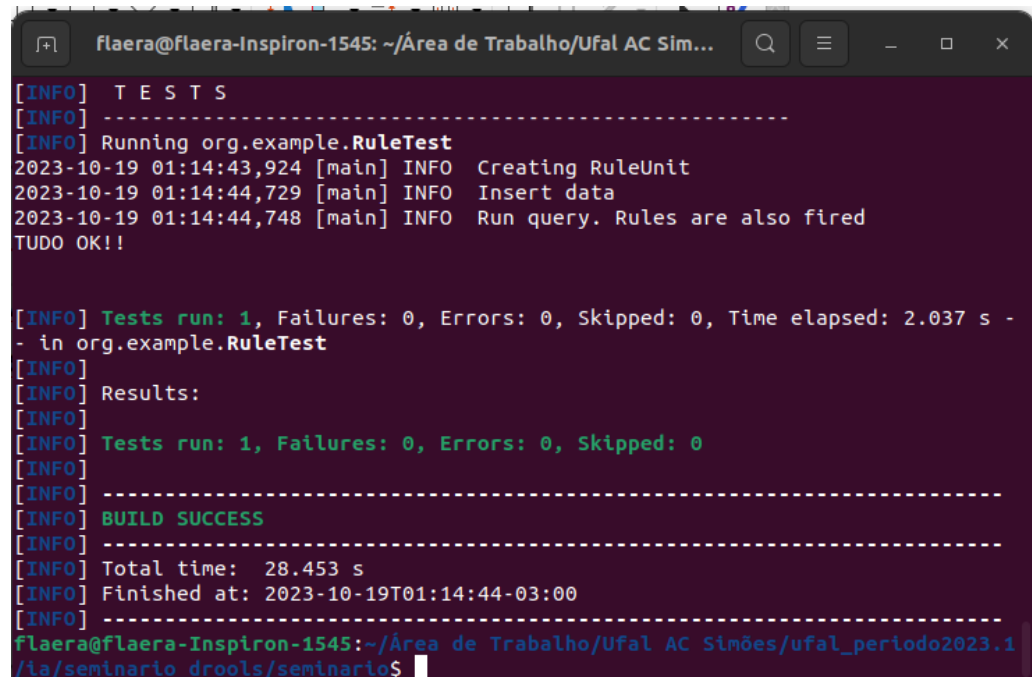
Conhecendo as Regras

seminario > src > main > resources > org > example > ≡ rules.drl

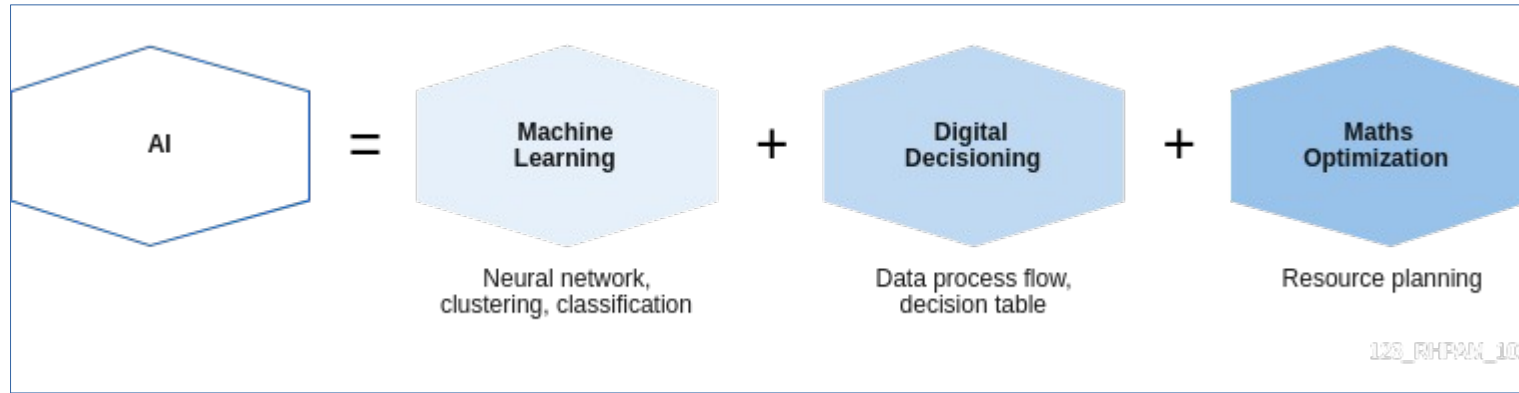
```
1
2 package org.example;
3
4 unit MeasurementUnit;
5
6 rule "will execute per each Measurement having ID color"
7 when
8 |   /measurements[ id == "color", $colorVal : val ]
9 then
10 |   controlSet.add($colorVal);
11 end
12
13 rule "red"
14 dialect "java"
15 when
16 |   $red : Measurement(getVal()=="red")
17 then
18 |   System.out.println("RED: "+$red.getVal());
19 end
20
21 query FindColor
22 |   $m: /measurements[ id == "color" ]
23 end
24
```

Build

- Execução via servidor Maven Apache.
- Comando: “*mvn clean test*”.
- Testes feitos em Linux Plataforma.

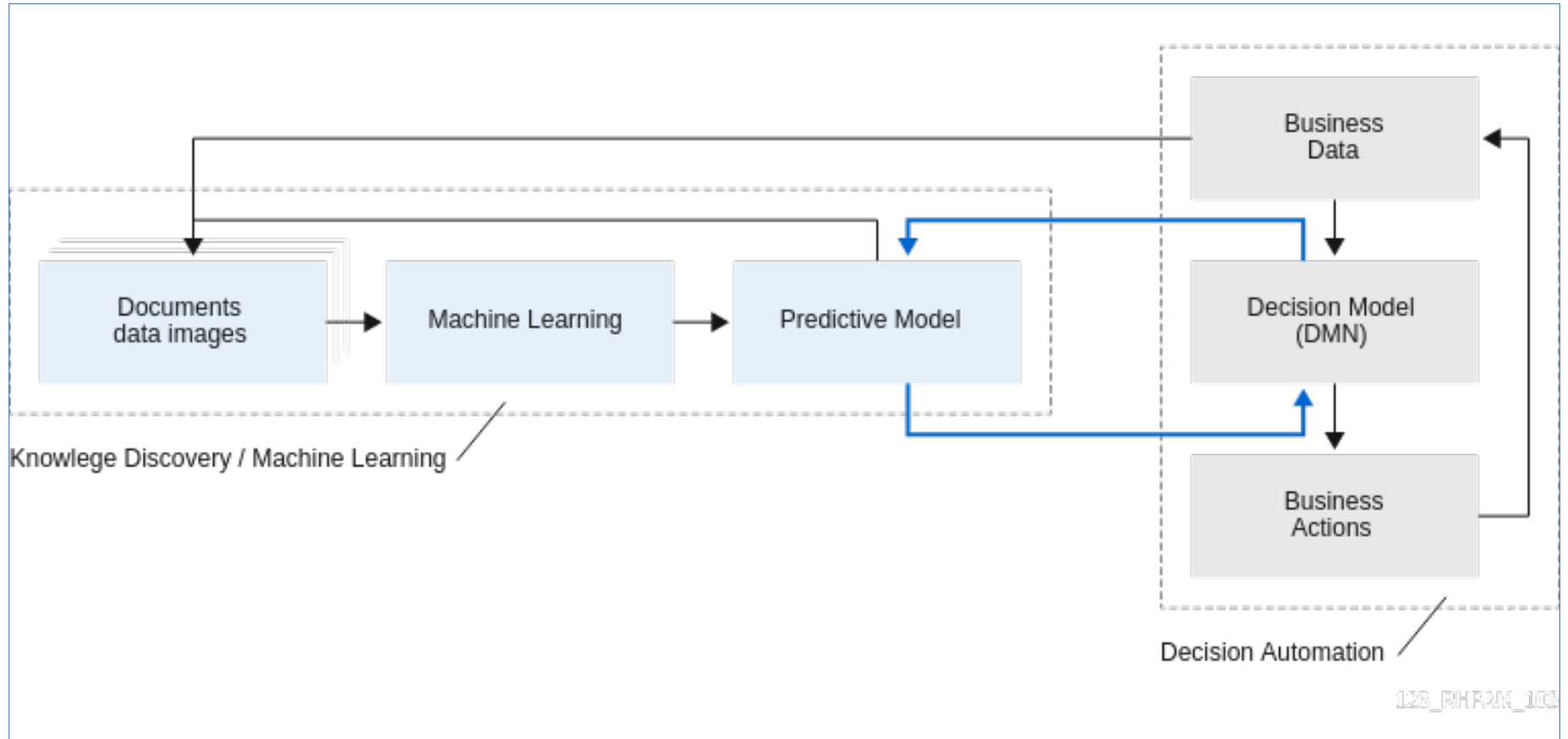
A terminal window with a dark purple background and light green text. The window title is 'flaera@flaera-Inspiron-1545: ~/Área de Trabalho/Ufal AC Sim...'. The output shows the execution of 'mvn clean test'. It starts with '[INFO] T E S T S' followed by a separator line. Then '[INFO] Running org.example.RuleTest'. Log messages show 'Creating RuleUnit', 'Insert data', and 'Run query. Rules are also fired'. A summary line states 'TUDO OK!!'. Another summary line shows 'Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 2.037 s - in org.example.RuleTest'. This is followed by '[INFO] Results:' and another summary line 'Tests run: 1, Failures: 0, Errors: 0, Skipped: 0'. A final separator line is followed by '[INFO] BUILD SUCCESS'. The last summary line shows 'Total time: 28.453 s' and 'Finished at: 2023-10-19T01:14:44-03:00'. The prompt at the bottom is 'flaera@flaera-Inspiron-1545:~/Área de Trabalho/Ufal AC Simões/ufal_periodo2023.1/ia/seminario_drools/seminario\$'.

Pragmática IA e Drools em ML



- Business Process Model and Notation (BPMN2).
- Decision Model and Notation (DMN).
- Predictive Model Markup Language (PMML).

Pragmática IA e Drools em ML



Pragmática IA e Drools em ML

- Informações do negócio entra no sistema, por exemplo, informações da aplicação de empréstimo.
- O modelo de decisão que é integrado com um preditivo modelo que decide quando ou não aprovar o empréstimo ou quando adicional tarefa é requerida.
- Uma ação de negócios é resultante, por exemplo, uma carta de rejeição ou oferta de empréstimo é enviada para o cliente.

Pragmática IA e Drools em ML

“This use case example is about the fictitious Fortress Bank, the bank’s customer Joe, and the business process management (BPM) developer Michelle. First we will look at how the bank originally used AI through Drools digital decisioning and then we will see how Michelle enhanced the decision model with a Predictive Model Markup Language (PMML) model created from machine learning”. -

Documentation of Drools in Pragmatic AI: Integrating Machine Learning with Drools Session.

Referências bibliográficas

- Site do Drools. Link: <https://www.drools.org/>; acesso em 18/10/2023 às 23:00.
- Documentação do Drools – Running Session. Link: https://docs.drools.org/7.11.0.Final/drools-docs/html_single/#_running; acesso em 18/10/2023 às 22:53.
- Documentação do Drools – Getting Start Session. Link: <https://docs.drools.org/8.44.0.Final/drools-docs/drools/getting-started/index.html>; acesso em 18/10/2023 às 22:54.
- Documentação do Drools – Pragmatic IA: Integrating Machine Learning with Drools. Link: <https://docs.drools.org/8.44.0.Final/drools-docs/drools/pragmatic-ai/index.html>; acesso em 19/10/2023 às 04:30.
- Slide sobre Drools de Vinícius Maia de Holanda. Obtido via solicitação de material ao professor. Acesso em via Gmail institucional do IC (Instituto de Computação) em 18/10/2023 às 18:51.
- Drools para gerenciar regras de negócio - Youtube. Link: <https://www.youtube.com/watch?v=h2AzmuVFKfQ&t=612s>; acesso em 13/10/2023 às 16:00.
- Regras de negócios com Drools: Atualização em tempo real com KieScanner - YouTube. Link: <https://www.youtube.com/watch?v=hjZPXFTM-OM>; acesso em 13/10/2023 às 16:20.
- Basic components Drools rule engine – YouTube. Link: <https://www.youtube.com/watch?v=iS2a2EDymvY>; acesso em 19/10/2023 às 00:40.
- Método createStore() do Drools. Link: <https://docs.drools.org/8.31.0.Final/kie-api-javadoc/org/drools/ruleunits/api/DataSource.html>; acesso em 19/10/2023 às 05:15.



Obrigado pela atenção!!