Berger Levrault

Introduction à Moose

–

Benoît Verhaeghe

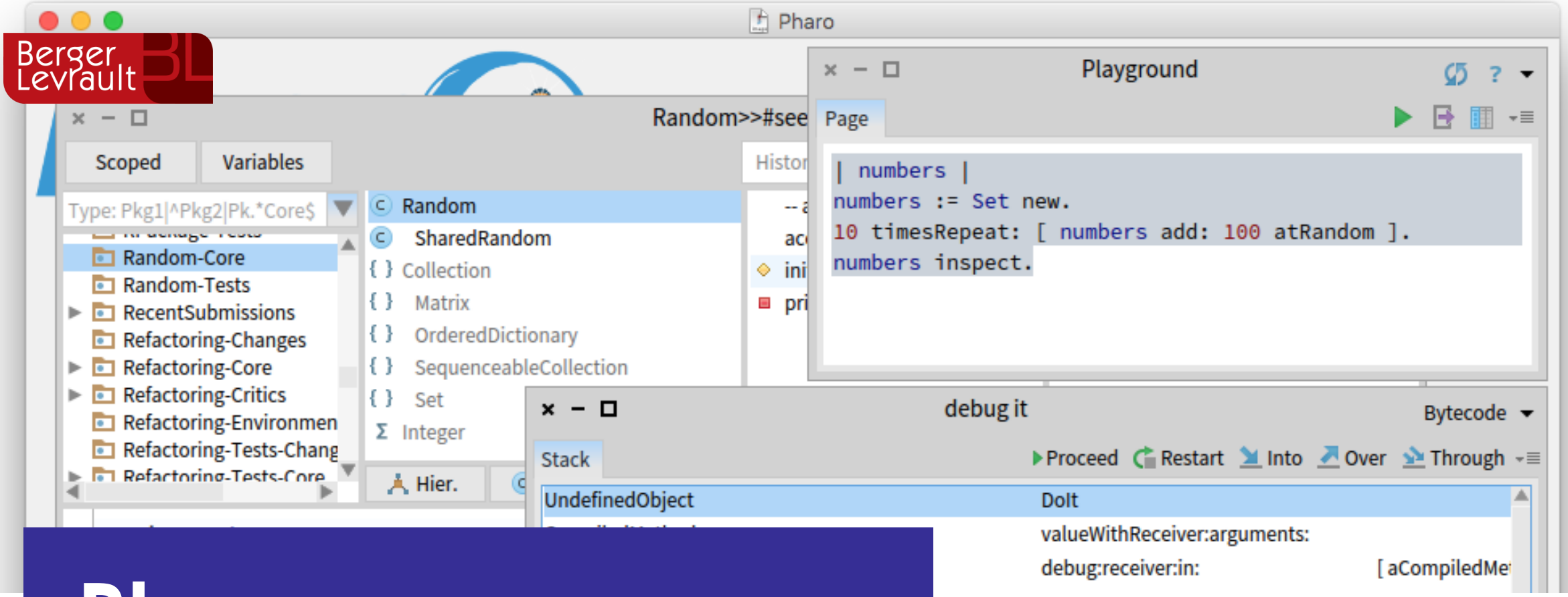
Merci !!!

Ces diapos sont inspirés de ceux de
Anne Etien
Nicolas Anquetil

Berger Levrault

Sommaire

- Pharo
- Visualisation
- Moose
- Qualité Logicielle
- Interface Graphique



Pharo

The immersive programming experience

Berger Levrault

Seulement des Objets

Messages

01

Unaire

Un message sans argument

42 factorial

02

Binaire

Un message avec un seul argument

41 + 1

03

Mot clefs

Un message avec plusieurs arguments

kill: ‘Cora’ with: aKnife

De Java à Pharo

```
String function(int param1, int param2){
    this.otherFunction(param1, param2);
    return “Hello World”;
}
```

De Java à Pharo

Enlever parenthèses, points, virgule, point-virgule

```
String function int param1 int param2

    this otherFunction param1 param2

    return “Hello World”
```

De Java à Pharo

Enlever les indicateurs des types

```
function param1 param2
    this otherFunction param1 param2
    return “Hello World”
```

De Java à Pharo

Remplacer par la nouvelle syntaxe
this → self
return → ^
“” → ‘’

```
function param1 param2

    self otherFunction param1 param2

    ^ ‘Hello World’
```

De Java à Pharo

Ajouter mots-clefs pour les fonctions et les points en fin de lignes

```
function: param1 with: param2
    self otherFunction: param1 with: param2
    ^ 'Hello World'
```

Résultat

```
String function(int param1, int param2){
    this.otherFunction(param1, param2);
    return “Hello World”;
}
```

```
function: param1 with: param2
    self otherFunction: param1 with: param2
    ^ ‘Hello World’
```

Berger Levrault

Visualisation

Avec Mondrian

Créer une view

```
view := RTMondrian new.
^ view
```

Ajouter des noeuds

```
view := RTMondrian new.
view nodes: { 1. 2. 3. 4. 5. }.
^ view
```

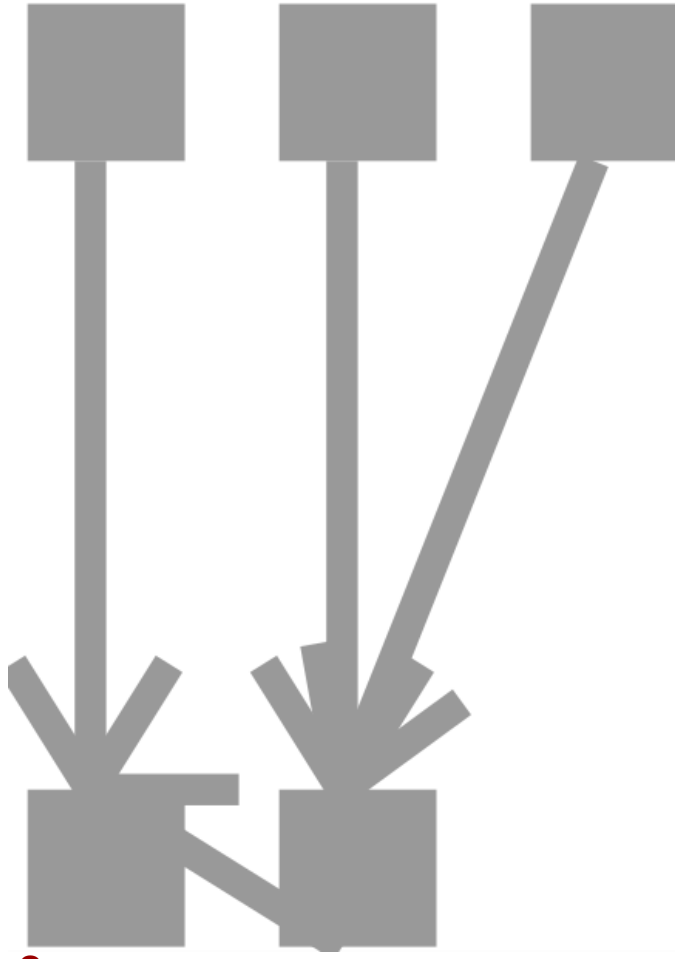
Ajouter des liens

```
view := RTMondrian new.
view nodes: { 1. 2. 3. 4. 5. }.
view edges useAssociations: { 1->2. 3->2. 4->3. 5->3 }.
^ view
```

Disposer les éléments

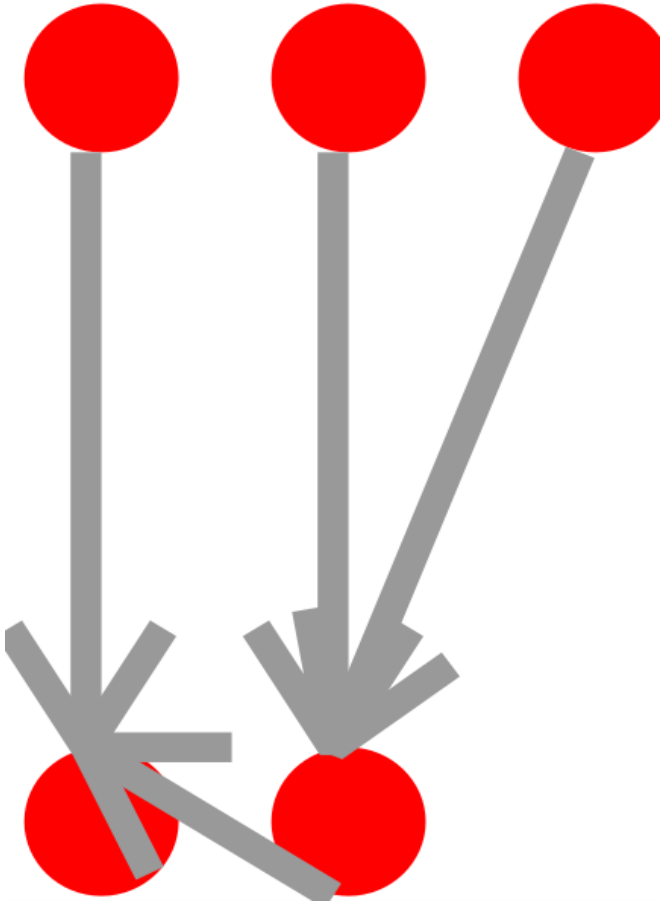
```
view := RTMondrian new.
view nodes: { 1. 2. 3. 4. 5. }.
view edges useAssociations: { 1->2. 3->2. 4->3. 5->3 }.
view layout tree.
^ view
```

Forme des liens



```
view := RTMondrian new.
view nodes: { 1. 2. 3. 4. 5. }.
view shape line arrowedLine shape head baseSize: 5.0; size: 8.
view edges useAssociations: { 1->2. 3->2. 4->3. 5->3 }.
view layout tree.
^ view
```

Forme des noeuds

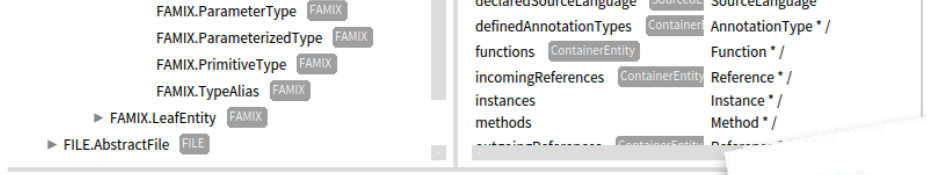


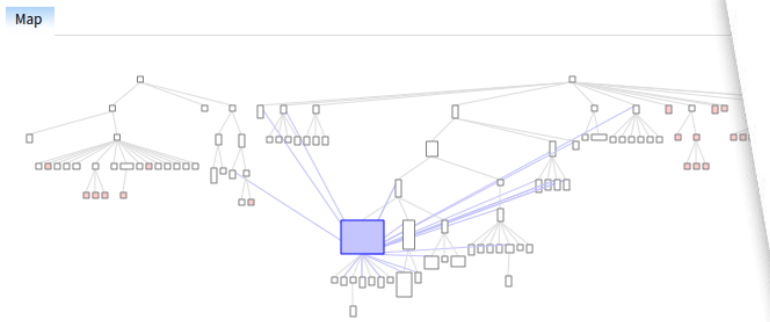
```
view := RTMondrian new.
view shape circle color: Color red.
view nodes: { 1. 2. 3. 4. 5. }.
view shape line arrowedLine shape head baseSize: 5.0; size: 8.
view edges useAssociations: { 1->2. 3->2. 4->3. 5->3 }.
view layout tree.
^ view
```

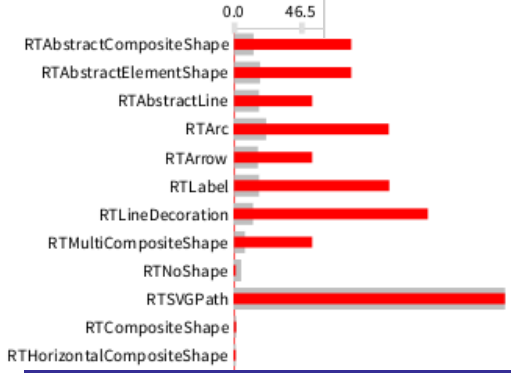
Forme des noeuds

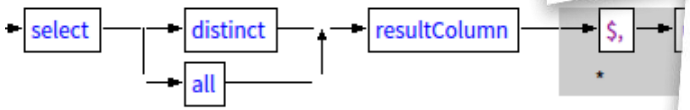

```
view := RTMondrian new.
view shape rectangle;
    height: [:class | class methods size];
    width: [:class | class methods sum: #numberOfLinesOfCode];
    withTextAbove: #name.
view nodes: {RTCalendarBuilder. RTMondrian.}.
^ view
```

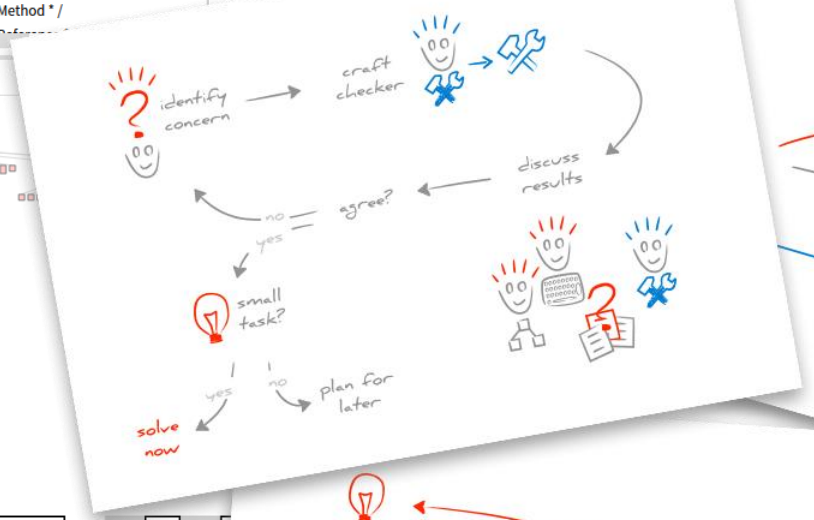
Berger Levrault

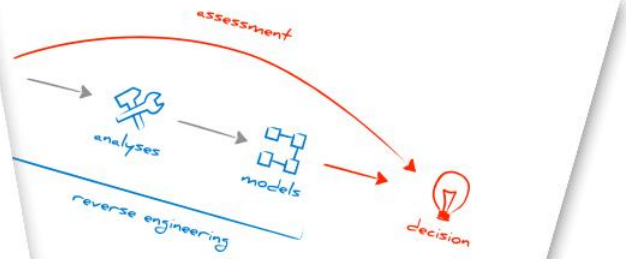


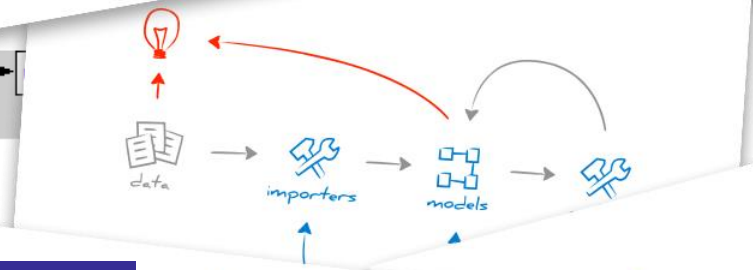


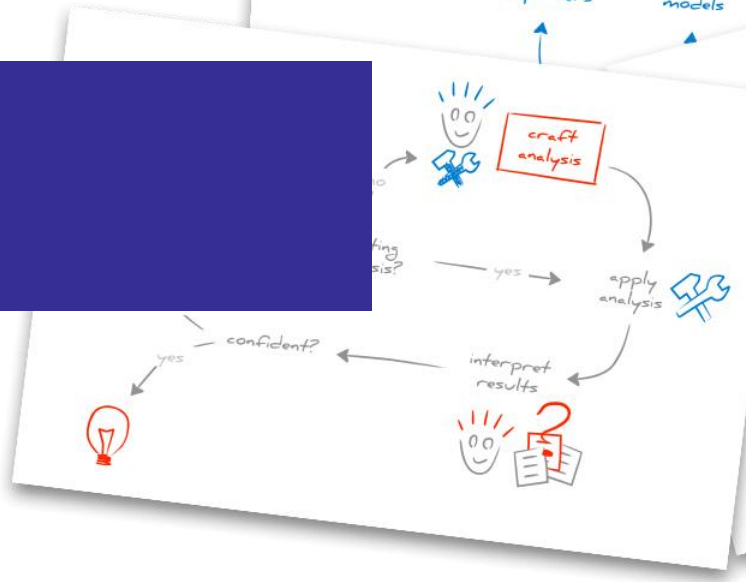


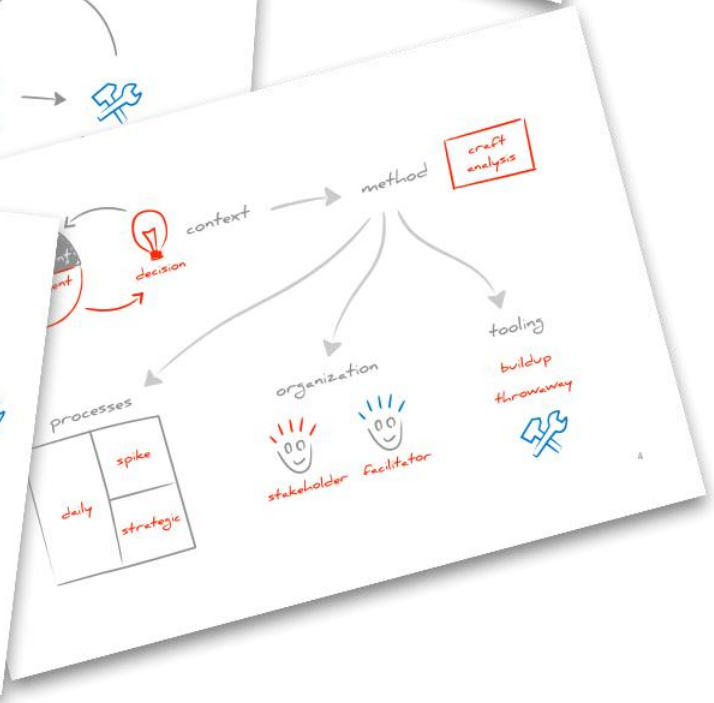






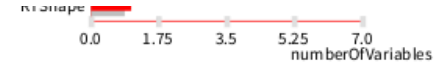




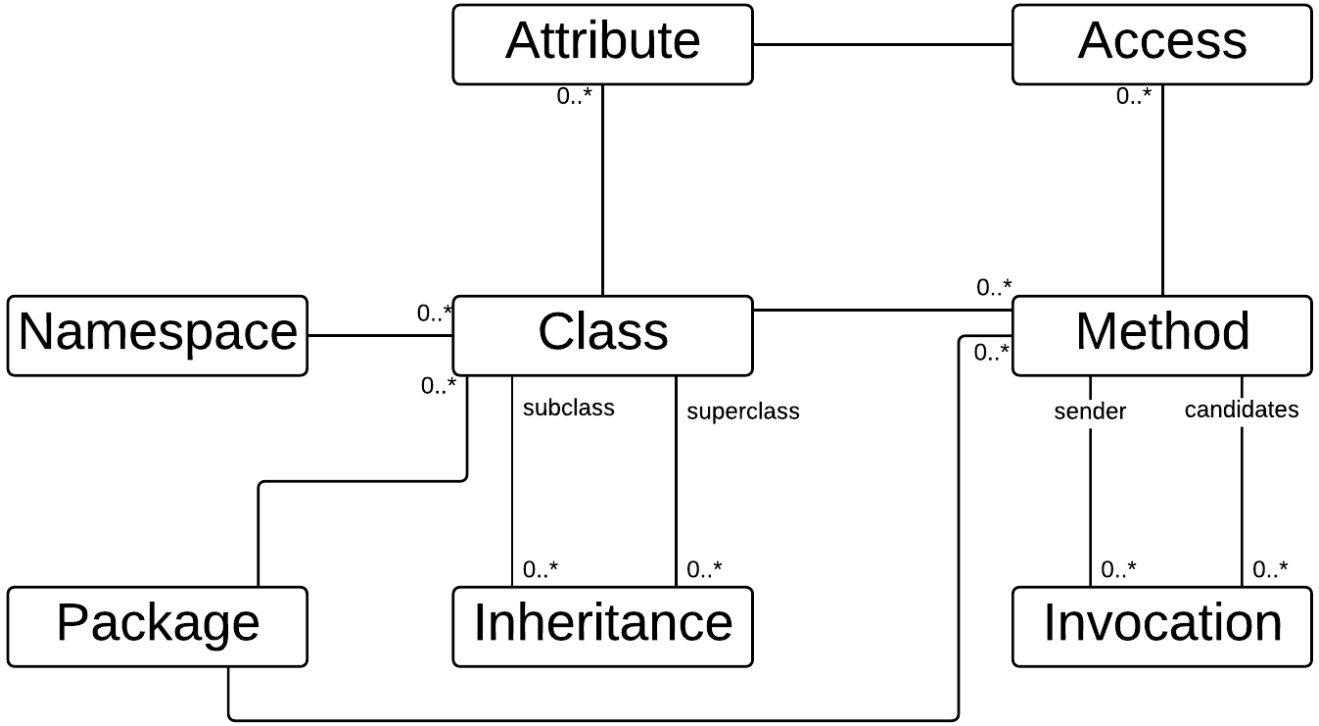


Moose

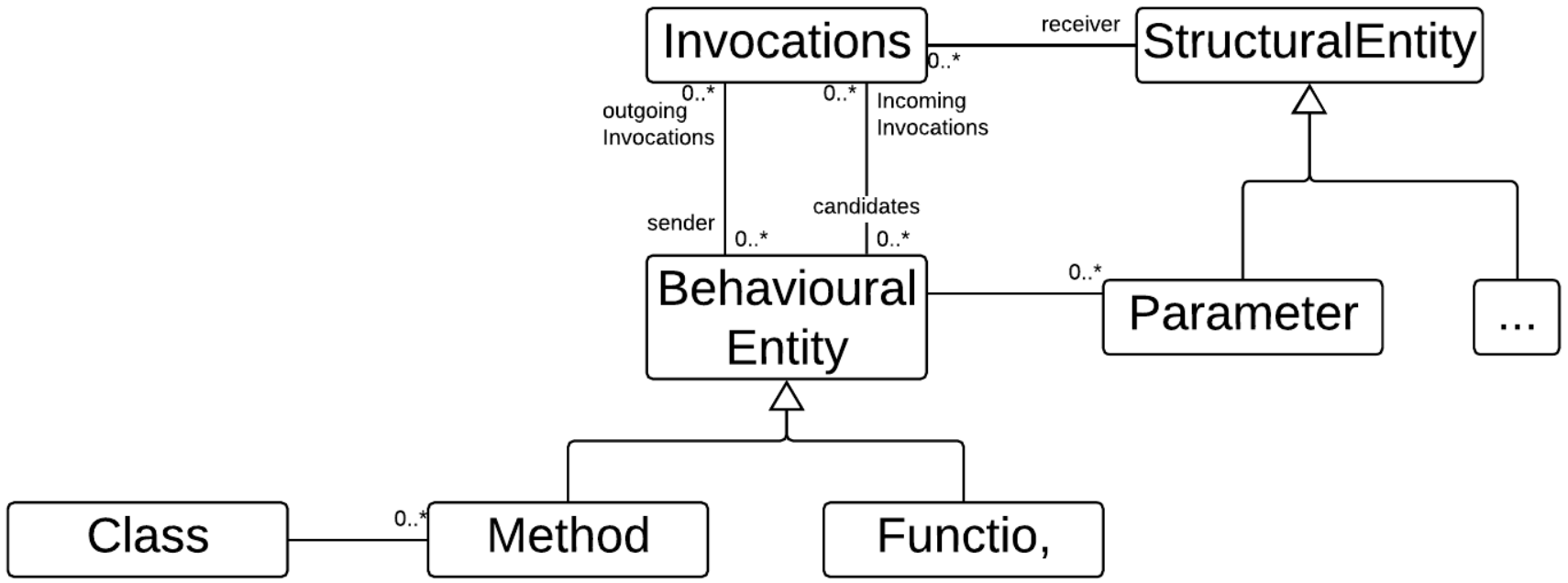
Famix Java



Modèle


Attribute
Access
0..*
0..*
Namespace
Class
Method
0..*
0..*
0..*
0..*
subclass
superclass
sender
candidates
0..*
0..*
0..*
0..*
Package
Inheritance
Invocation

Invocations



Berger Levrault

Qualité Logicielle

Que pouvons nous étudier ?

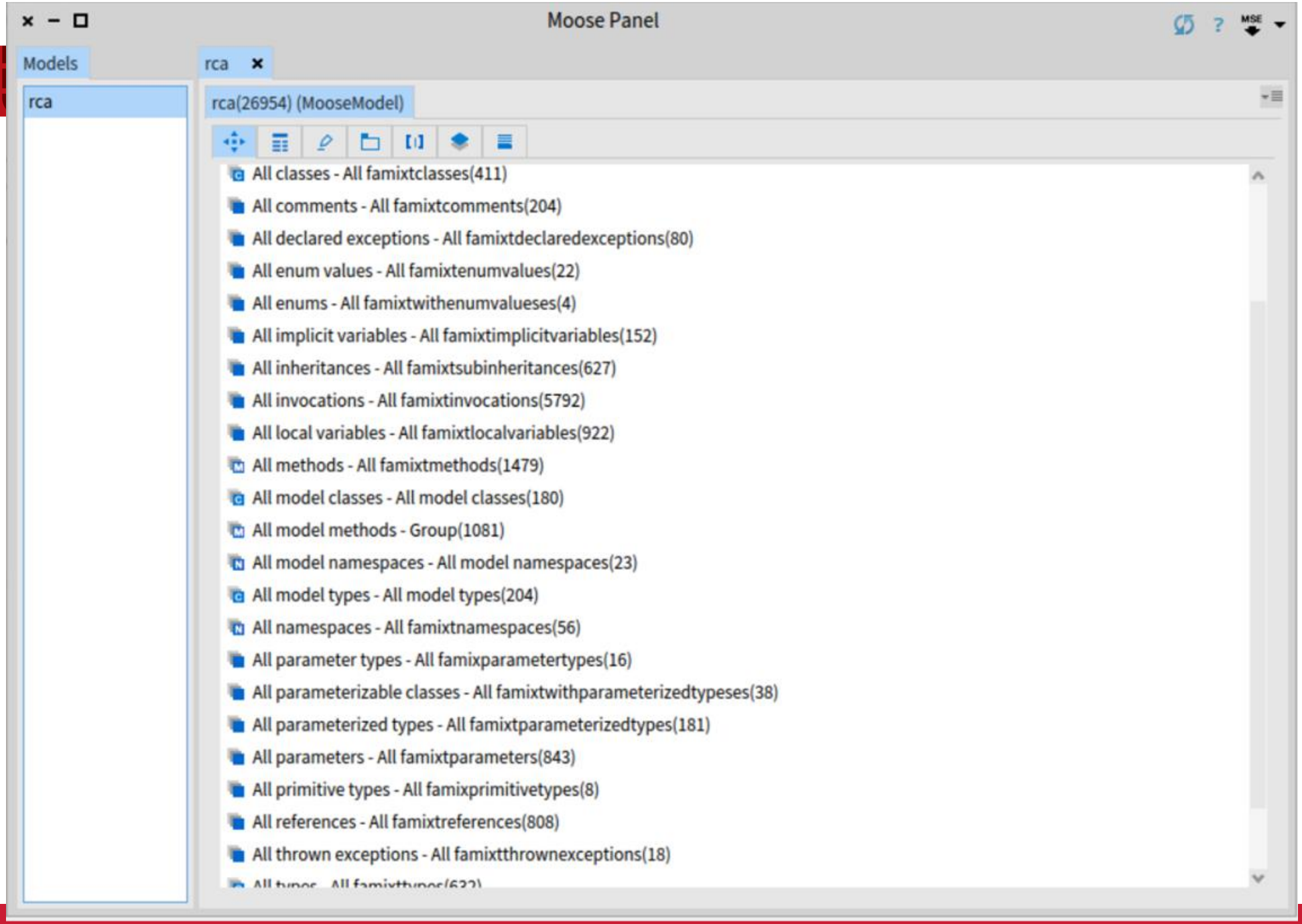
- Hiérarchie de paquetages
- Diagramme de classes
- Cycles entre paquetages
- Répartitions classes/paquetages
- God classes (LOC et nb de méthodes)
- Lazy classes (LOC et nb de méthodes)

- Nb de packages
- Nb de classes
- Nb de méthodes
- Nb d'attributs
- Code mort
- Tests
- Méthodes dépréciées
- Complexité cyclomatique

Import

```
mooseModel := MooseModel importFromMSEStream:
    msePath asFileReference readStream.
mooseModel rootFolder: rootFolderPath.
mooseModel name: 'rca'.
mooseModel install.
```

Berger Levrault



Paquetages/Classes/Méthodes

```
mooseModel allModelNamespaces
mooseModel allModelClasses
mooseModel allModelMethods
```

Dans le projet RCA

```
mooseModel allX
```

Dans le projet RCA
+
Les dépendances extérieures

Compter le nombre d'éléments

```
"Nombre de classes"
mooseModel allModelClasses size. "180"

"Nombre de lignes de code"
mooseModel allModelClasses sum: #numberOfLinesOfCode. "11613"

"Nombre de lignes de code - aussi"
mooseModel allModelClasses sum:
    [ :class | class numberOfLinesOfCode]. "11613"
```

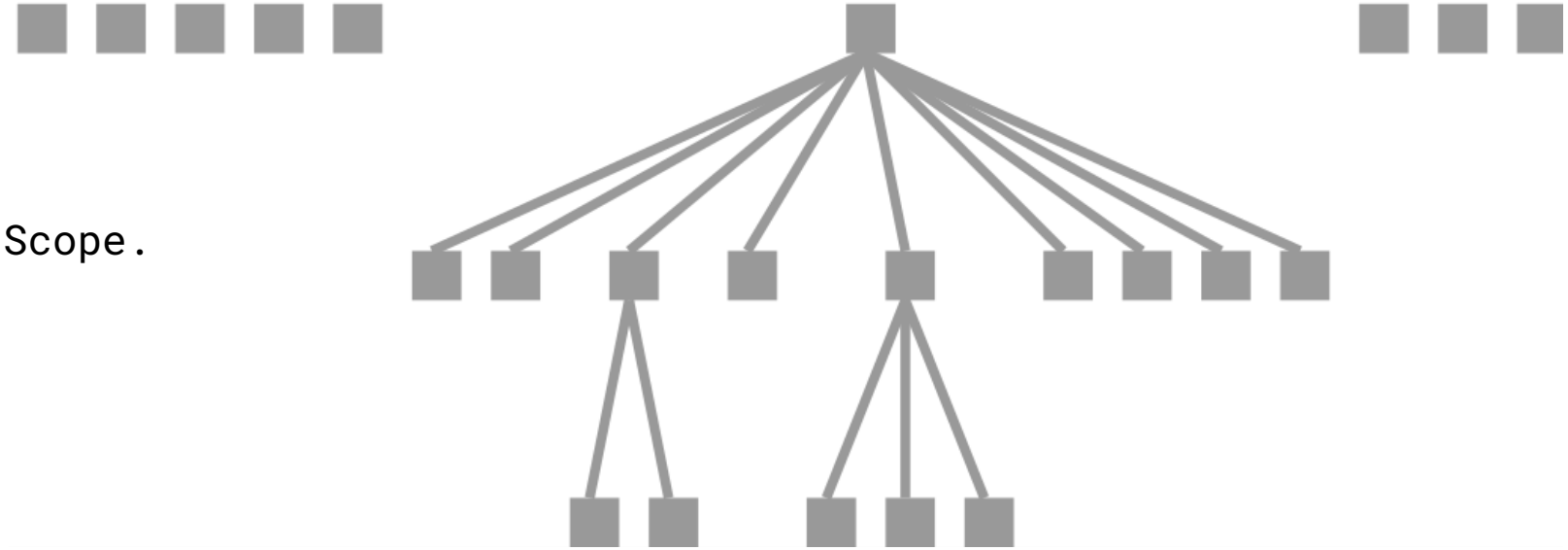
Chercher des annotations

```
“Sélectionner les classes avec une annotation”
mooseModel allModelClasses
    select: [:class | class annotationInstances isNotEmpty ].

“Sélectionner les classes dépréciées”
(mooseModel allModelClasses select: [:class |
    class annotationInstances isNotEmpty
      and: [ class annotationTypes anySatisfay:
        [:type | type name = ‘Deprecated’ ]]).
```

Hiérarchies de paquetages

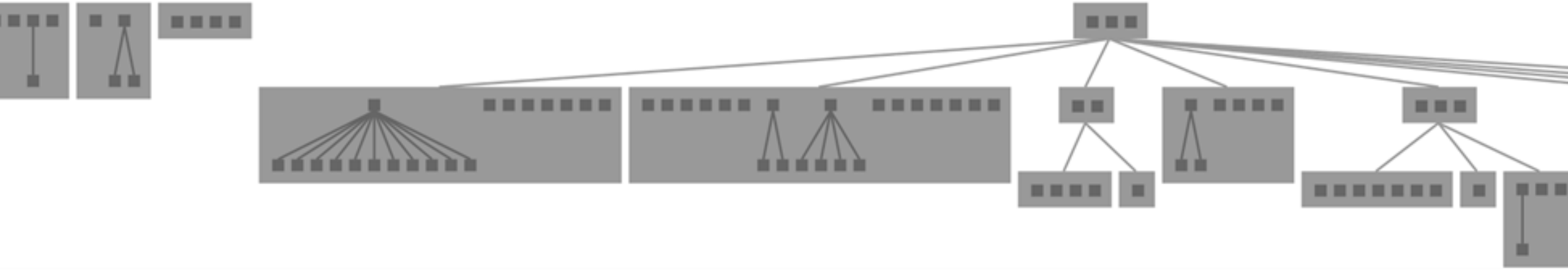
```
| b |
b := RTMondrian new.
b nodes: mooseModel
allModelNamespaces.
b edgesFrom: #parentScope.
b layout tree.
b build.
b view
```



Relations classes/paquetages

```
| b |
b := RTMondrian new.
b shape rectangle.
b nodes: model allModelNamespaces forEach: [ :p |
    b nodes: p classes.
    b edges connectFrom: #superclass.
   b layout tree ].
b edgesFrom: #parentScope.
b layout tree.
b build.
^ b view
```

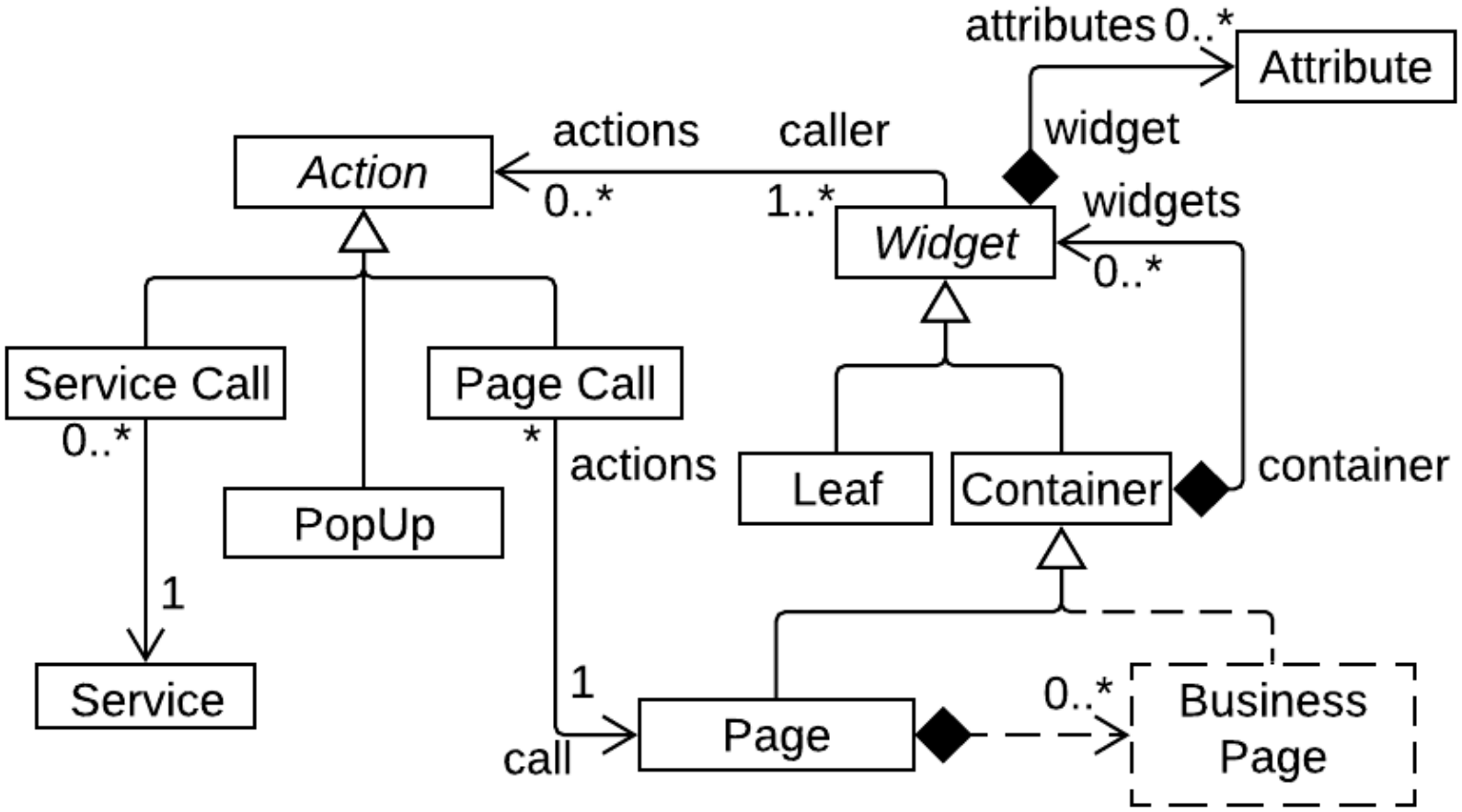
Relations classes/paquetages



Berger Levrault

GUI Example

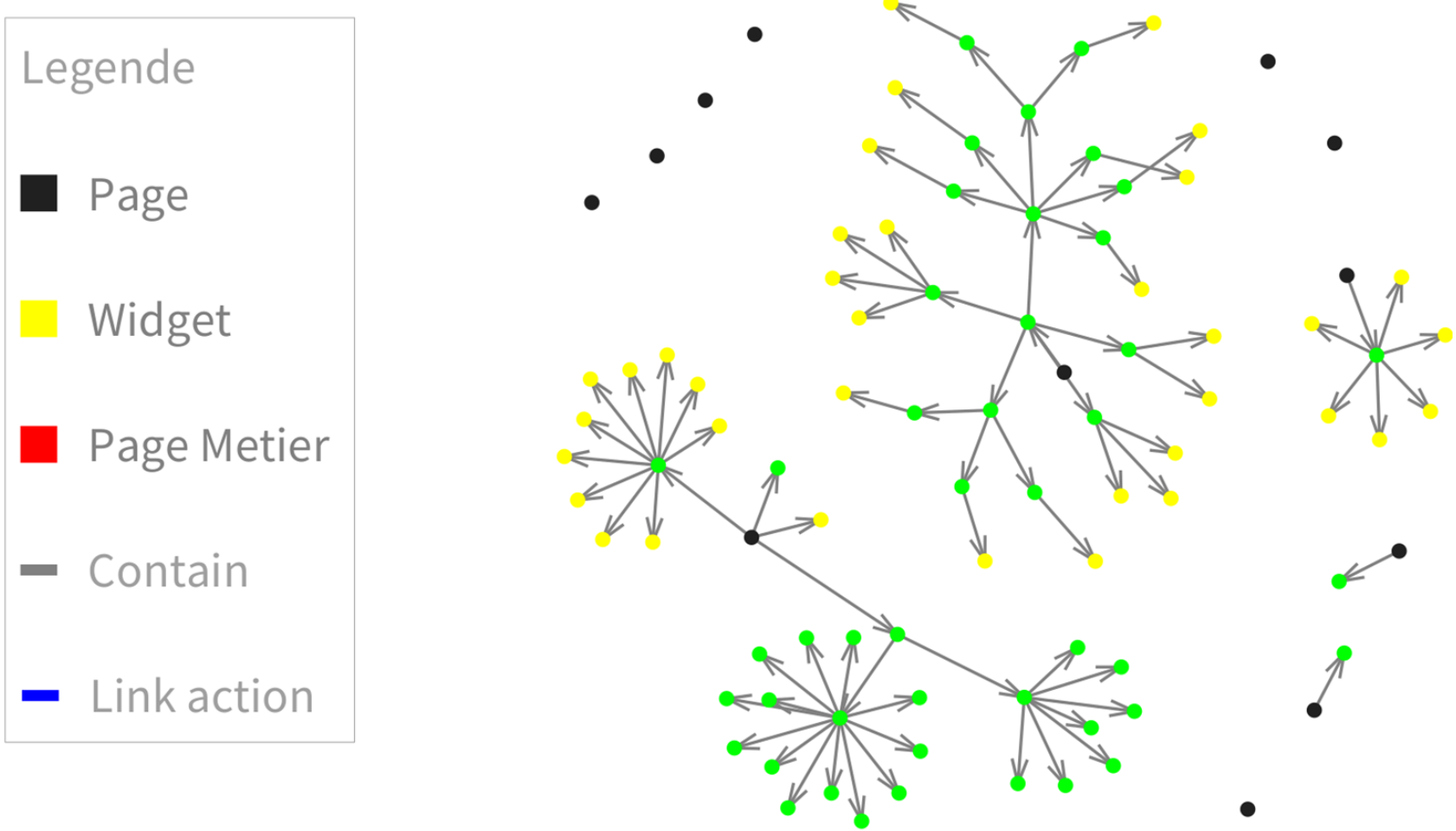
Comment représenter une Interface Graphique ?



Visualisation générale

```
| b |
b := RTMondrian new.
b shape circle color: [ :cl | cl color ].
b nodes: (aBLModel allBLWidget , aBLModel allBLService) asSet
    asOrderedCollection.
(b shape line arrowedLine
   headOffset: 1.0;) shape head
   baseSize: 5.0;
   size: 8.
b edges useAssociations: (aBLModel allBLWidget collect: #allAssociations) flatten.
b layout force strength: 0.4.
b view pushBackEdges.
^ b
```

Et pour RCA ?



Berger Levrault

TP

https://badetitou.github.io/teaching/mtp/VisuRCA/

Visualisations de Modèles

berger-levrault.com