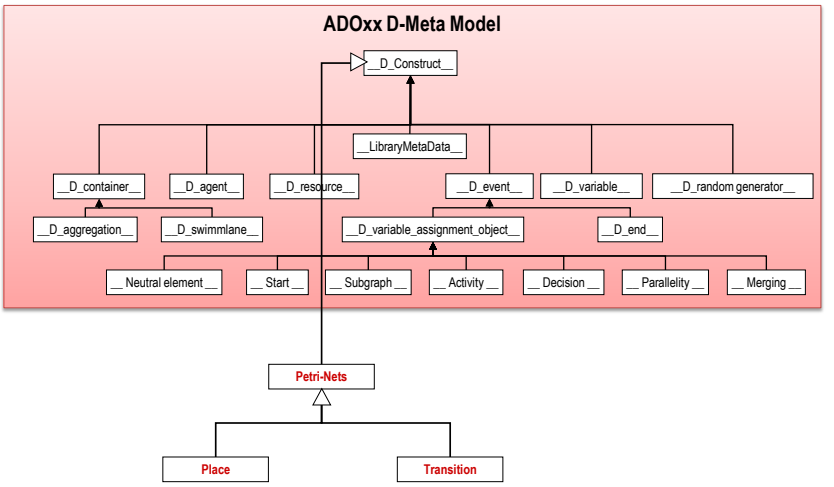
IMPLEMENTATION OF A SIMULATION ALGORITHM ON A MODELLING LANGUAGE

Motivation

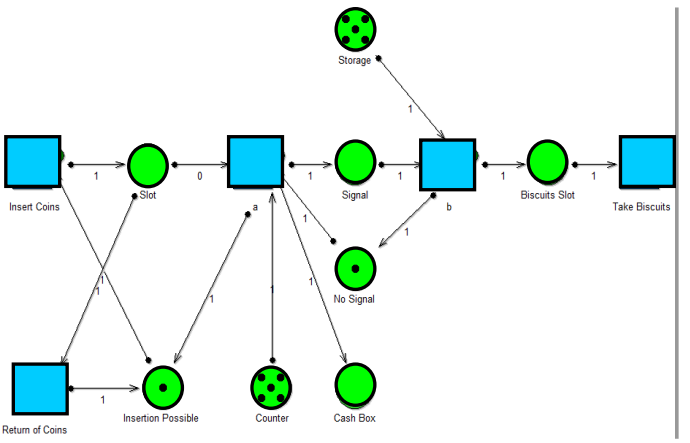
Create a Petri Nets Modelling Method that allow;

- Defining a ***System***
- Defining ***Active States*** of the system
- Defining ***Passive States***
- Defining ***Relations*** between active states and passive states
- Defining ***Initial Resource Distribution***
- ***Simulation*** on defined system

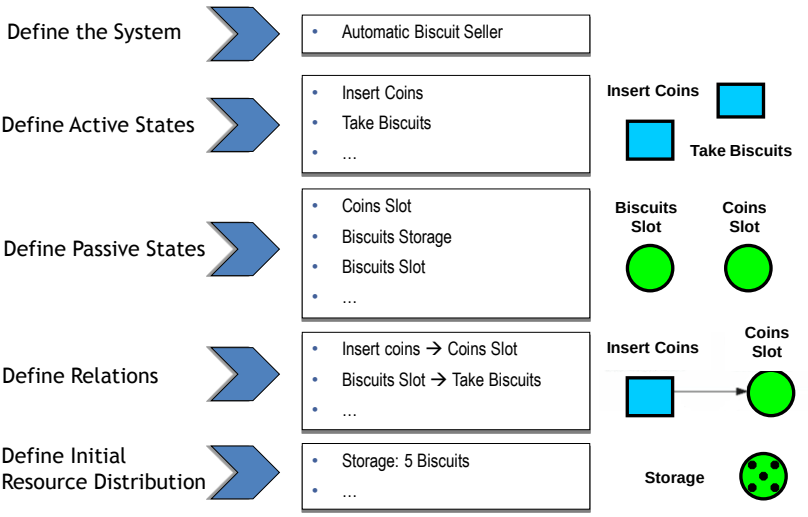
Inheriting of Petri Nets-Meta Model from ADOxx D-Meta Model



Sample System: Automatic Biscuits Seller



Overview



Create Classes

Scenario: Create classes namely “Petri-Nets”, “Place” and “Transition”

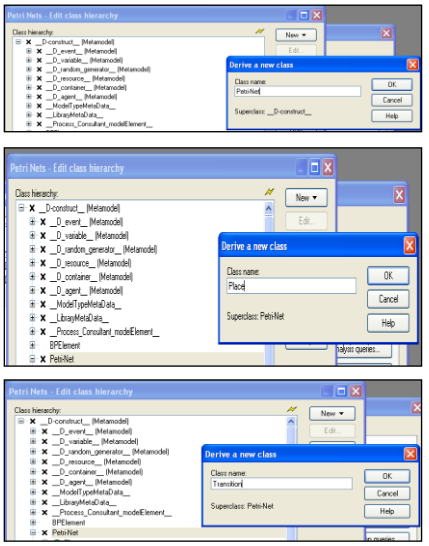
1. Open the „Class hierarchy“ for the Dynamic library.

2. Activate the “Metamodel” then “class hierarchy” .

3. Add the class „Petri-Nets“ derived from the super class „__D-construct__“.

4a. Add the class „Place“ derived from the super class „Petri-Nets“.

4b. Add the class „Transition“ derived from the super class „Petri-Nets“.



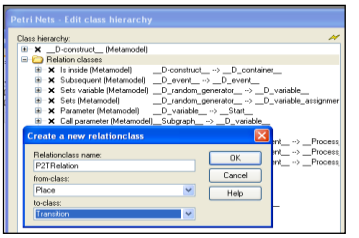
Create Relation Classes

Scenario: Create relation classes namely “P2TRelation” and “T2PRelation”

1. Open the „Class hierarchy“ for the Dynamic library.

2. Activate the “Metamodel” then “class hierarchy”

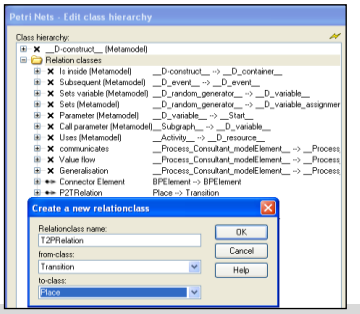
3a. Add the relation class „P2TRelation“ under „Relation classes“



4a. Assign class “Place” as “from-class” and class “Transition” as “to-class”

3b. Add the relation class „T2PRelation“ under „Relation classes“

4b. Assign class “Transition” as “from-class” and class “Place” as “to-class”



Create Attributes of Class “Place”

Scenario: Create attributes of class “Place” namely “ClassName”, “NumOfTokens” and in type of “STRING” , “INTEGER” respectively.

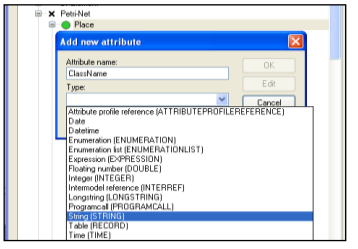
1. Open the „Class hierarchy“ for the Dynamic library.

2. Activate the “Metamodel” then “class hierarchy”

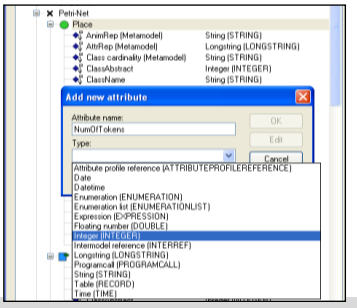
3. Select the class “Place”

4. Click “New attribute”

5a. Create attribute “ClassName” in type of “STRING”



5b. Create attribute “NumOfTokens” in type of “INTEGER”



Create Attributes of Class “Transition”

Scenario: Create attributes of class “Transition” namely “ClassName”, “GameEnabled” and “State” in type of “STRING” , “INTEGER” and “ENUMERATION” respectively. Enumeration domain of “State” is “Cold or Hot”.

1. Open the „Class hierarchy“ for the Dynamic library.

2. Activate the “Metamodel” then “class hierarchy”

3. Select the class “Transition”

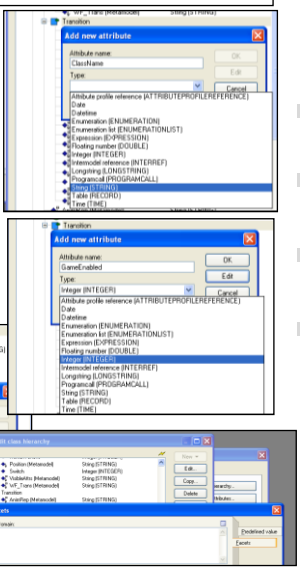
4. Click “New attribute”

5a. Create attribute “ClassName” in type of “STRING”

5b. Create attribute “GameEnabled” in type of “INTEGER”

5c. Create attribute “State” in type of “ENUMERATION”

6c. Click edit, select “Facets” and add value “Cold@Hot” in “EnumarationDomain”



Create Attribute of Relations “P2TRelation” and “T2PRelation”

Scenario: Create attributes of relation classes “P2TRelation” and “T2PRelation” namely “Weight”, “Name”, “Visible” in type of “ “INTEGER”, “STRING” and “ENUMERATION” respectively.

1. Open the „Class hierarchy“ for the Dynamic library.

2. Activate the “Metamodel” then “class hierarchy”

3. Select the relation class “P2TRelation””

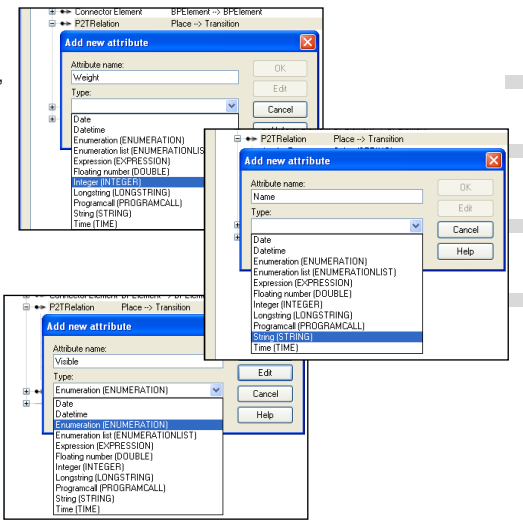
4. Click “New attribute”

5a. Create attribute “Weight” in type of “INTEGER”

5b. Create attribute “Name” in type of “STRING”

5c. Create attribute “Visible” in type of “ENUMERATION”, select “Facets” and add value “Yes@No” in “EnumarationDomain”

6. Repeat steps 3-5b for relation class “T2PRelaiton”



Graphical Representation of Place

Scenario: Tokens shall be shown as black ellipses as many as the number defined in attribute "NumOfTokens" until the number smaller than 6, else they shall be shown in form of number

```
GRAPHREP
PEN w:0.05cm
FILL color:limegreen
ELLIPSE 3 x1:-.7cm rx:.7cm ry:.7cm
ATTR "Name" y:.8cm w:c:2.8cm h:t
AVAL no:"NumOfTokens"
```

```
IF ( no = "0" )
FILL color:limegreen
ELLIPSE 4 x1:-.7cm rx:.7cm ry:.7cm
```

Place

```
ELSIF ( no = "1")
FILL color:black
ELLIPSE x:0cm rx:.15cm ry:.15cm
```

Place-3

```
ELSIF ( no = "2")
FILL color:black
ELLIPSE x:-.2cm rx:.15cm ry:.15cm
ELLIPSE x:.2cm rx:.15cm ry:.15cm
```

Place-1

```
ELSIF ( no = "3")
FILL color:black
ELLIPSE x:-.2cm y:-.1cm rx:.15cm ry:.15cm
ELLIPSE x:.2cm y:-.1cm rx:.15cm ry:.15cm
ELLIPSE y:.3cm rx:.15cm ry:.15cm
```

Place-2

```
ELSIF ( no = "4")
FILL color:black
ELLIPSE x:-.27cm y:0cm rx:.15cm ry:.15cm
ELLIPSE x:.27cm y:0cm rx:.15cm ry:.15cm
ELLIPSE y:.3cm rx:.15cm ry:.15cm
ELLIPSE y:-.3cm rx:.15cm ry:.15cm
```

Place-4

```
ELSIF ( no = "5")
FILL color:black
ELLIPSE x:-.34cm y:0cm rx:.15cm ry:.15cm
ELLIPSE x:.34cm y:0cm rx:.15cm ry:.15cm
ELLIPSE y:.34cm rx:.15cm ry:.15cm
ELLIPSE y:-.34cm rx:.15cm ry:.15cm
ELLIPSE x:0cm y:0cm rx:.15cm ry:.15cm
```

Place-5

```
ELSE
ATTR "NumOfTokens" y:-.2cm w:c:2.8 cm h:t
ELLIPSE x:0cm y:0cm rx:.15cm ry:.15cm
```

6

Place-6

```
ENDIF
```


Graphical Representation of Transition

Scenario: Tokens shall be shown as black ellipses as many as the number defined in attribute "NumOfTokens" until the number smaller than 6, else they shall be shown in form of number

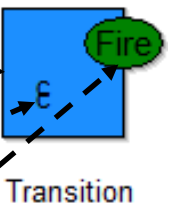
```
GRAPHREP
PEN w:0.05cm
```

```
FILL color: dodgerblue
RECTANGLE 3 x:-1.4cm y:-.7cm x2:.7cm w:1.4cm h:1.5cm
```

```
ATTR "Name" y:1.2cm x:-0.6cm w:c h:t
AVAL  s: "State"
```

```
IF (s="Cold")
TEXT "c" y:0cm x:-1.0cm h:t
TEXT "c" y:0.12cm x:-1.0cm h:t
ENDIF
```

```
FILL color:green
ELLIPSE x:0cm y:-0.3cm rx:0.45cm ry:0.3cm
FONT h:0.4cm
TEXT "Fire" x:0cm y:-0.26cm w:c h:c
```

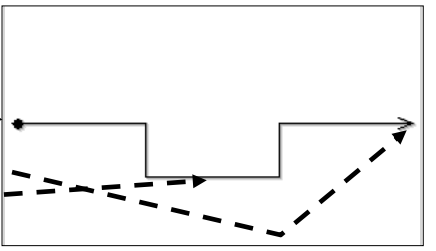


Graphical Representation of Relations “P2TRelation” and “T2PRelation”

Scenario: Represent at the start of relation black ellipse and at the end a arrow. If attribute “Visible” set as “Yes” show the weight of Relation.

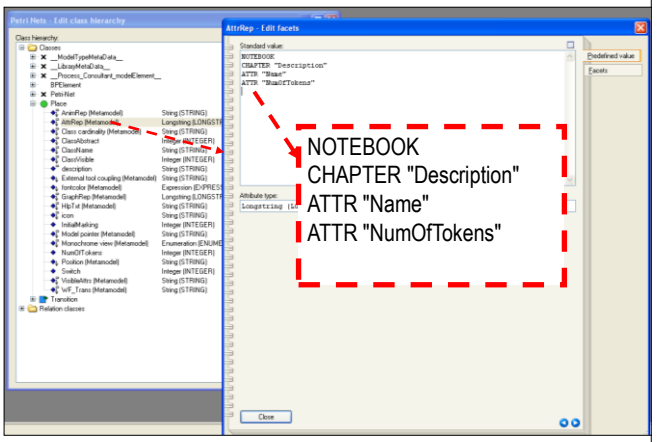
```
GRAPHREP
START
FILL color:black
ELLIPSE x:-.1cm rx:.1cm ry:.1cm
END
LINE x1:-.3cm y1:.1cm x2:0cm y2:0cm
LINE x1:-.3cm y1:-.1cm x2:0cm y2:0cm
MIDDLE

AVAL v:"Visible"
IF (v = "Yes")
ATTR "Weight" y:.4cm w:c h:c
ELSE
ENDIF
```



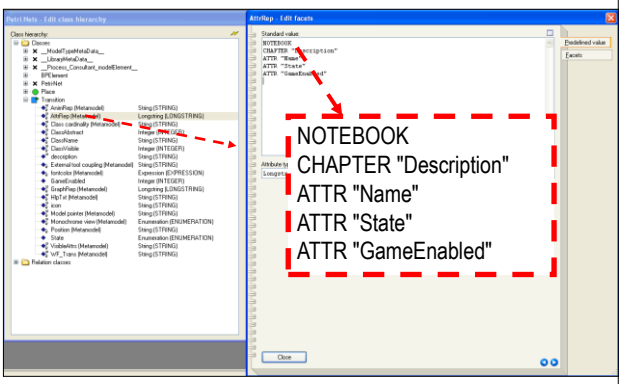
Notebook Definition of “Place”

1. Open attribute “AttrRep” of class “Place” to define the attribute representation
2. Write ATTRREP code to provide a notation for the class



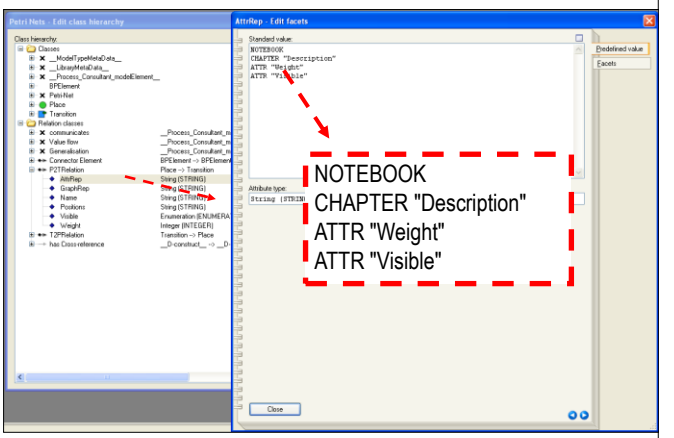
Notebook Definition of “Transition”

1. Open attribute “AttrRep” of class “Transition” to define the attribute representation
2. Write ATTRREP code to provide a notation for the class



Notebook Definition of “P2TRelation” and “T2PRelation”

1. Open attribute “AttrRep” of class “Transition” to define the attribute representation
2. Write ATTRREP code to provide a notation for the class



Simulation Algorithm

Simulation: execution of a Petri Nets (consumption /production of Tokens)

- **Fast Simulation**
- **Step by Step Simulation**

Fast simulation partial pseudo code:

```
Take number of iterations

For (number of iterations)
{
    For (number of transition)
    {
            Select randomly a transition

            Enable this transition for simulation

            Find all Place2TransitionArcs of this transition

                For (all Places which are related to this transition with
                Place2TransitionArc)
                {
                        If (number of tokens of Place >= weight of Place2TransitionArc)
                        {
                            Set number of tokens of Place: number of tokens of Place –
                            weight of Place2TransitionArc

                        }

                        Else
                        {
```


Simulation Algorithm

Get number of iteration

```
267  CC "AdoScript" EDITFIELD caption: "Iterations" title: "Number of Iterations"
268  SET executions: (VAL text)
269
```

Select transition randomly

```
280      FOR j from: 0 to: (numoftransitions -1)
281      {
282          # Random Selection
283          SET r: (0.0)
284          SET r: (random()*numoftransitions -1)
285          SET s: (INT ceil(r))
286          SET transitionId:(token(transitionIds, s, " "))
287
```

Enable selected transition

```
288          # Enable a Transition
289          CC "Core" SET_ATTR_VAL objid: (VAL transitionId) attrid:(isEnabledId) val:(1)
290          # Assign current value to the isEnabled variable
291          SET isEnabled: (1)
292
```

Simulation Algorithm

Choose all incoming relations of transition

```
293             # Choose all the incomming Relations
294             CC "Core" GET_CONNECTORS objid:(VAL transitionId) in
295             SET incommingFlowRelationsIds: (objids)
```

Check weight of if number of tokens in places from where incoming relation comes bigger that the weight of relation. If yes disable the transition.

```
297         FOR incommingFlowRelationsId in:(incommingFlowRelationsIds)
298         {
299             # Get the Ids of the EndPoints of the FlowRelations
300             CC "Core" GET_CONNECTOR_ENDPOINTS objid:(VAL incommingFlowRelationsId)
301
302             # Get the Number Of Tokens of the Place
303             CC "Core" GET_ATTR_VAL objid:(fromobjid) attrid:(numberOfTokenId)
304             SET placeNumberOfToken:(val)
305
306             # Get the weight of the incomming connector
307             CC "Core" GET_ATTR_VAL objid:(VAL incommingFlowRelationsId) attrid:(p2TWeightId)
308             SET incommingFlowRelationsWeight:(val)
309
310             IF(placeNumberOfToken < incommingFlowRelationsWeight)
311             {
312                 # Disable Transition
313                 CC "Core" SET_ATTR_VAL objid: (VAL transitionId) attrid:(isEnabledId) val:(0)
314                 # Assign the current value to the isEnabled variable
315                 SET isEnabled: (0)
316                 BREAK
317             }
```


Simulation Algorithm

Check weight of if number of tokens in places from where incoming relation comes bigger than the weight of relation. If yes consume tokens from the place.

```
327
328         FOR incommingFlowRelationsId in:(incommingFlowRelationsIds)
329         {
330             # Get the Ids of the EndPoints of the FlowRelations
331             CC "Core" GET_CONNECTOR_ENDPOINTS objid:(VAL incommingFlowRelationsId)
332
333             # Get the Number Of Tokens of the Place
334             CC "Core" GET_ATTR_VAL objid:(fromobjid) attrid:(numberOfTokenId)
335             SET placeNumberOfToken:(val)
336
337             # Get the weight of the incomming connector
338             CC "Core" GET_ATTR_VAL objid:(VAL incommingFlowRelationsId) attrid:(p2TWeightId)
339             SET incommingFlowRelationsWeight:(val)
340
341             IF(placeNumberOfToken >= incommingFlowRelationsWeight)
342             {
343                 # Delete the consumed Token
344                 CC "Core" SET_ATTR_VAL objid:(fromobjid) attrid:(numberOfTokenId) val:(placeNumberOfToken - incommingFlowRelationsWeight)
345
346             }
347         }
```

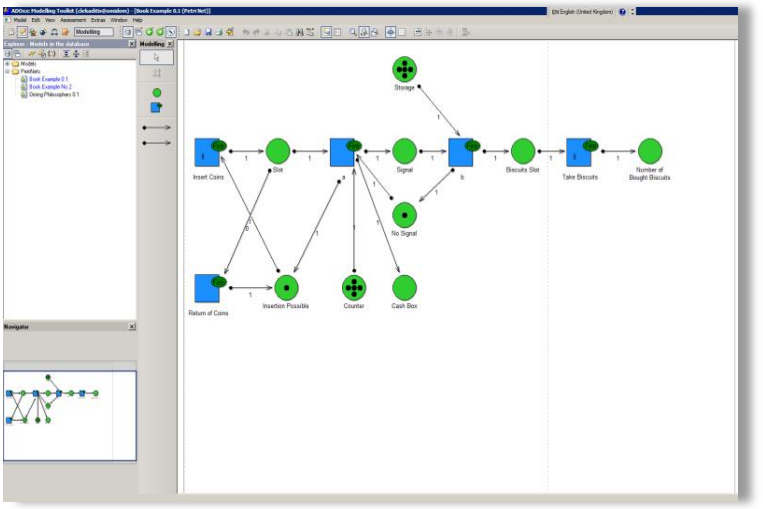
Simulation Algorithm

Choose all outgoing relation of transition

```
348
349                 # Choose all the outgoing Relations
350                 CC "Core" GET_CONNECTORS objid:(VAL transitionId) out
351
```

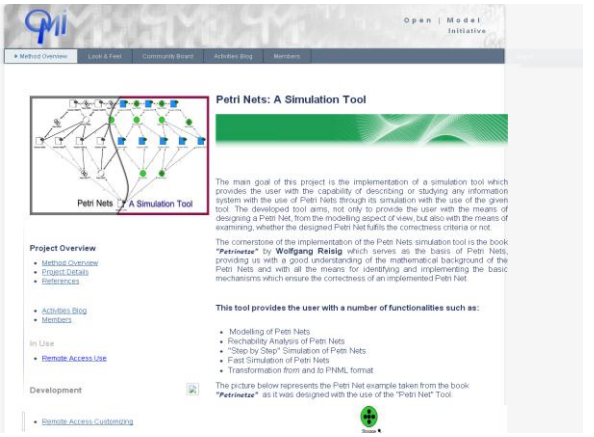
Add new tokens to places to where outgoing relations go.

```
349             # Choose all the outgoing Relations
350             CC "Core" GET_CONNECTORS objid:(VAL transitionId) out
351
352             # Add the newly produced Token
353             FOR outgoingFlowRelationsId in:(objids)
354             {
355                 # Get the weight of the incomming connector
356                 CC "Core" GET_ATTR_VAL objid:(VAL outgoingFlowRelationsId) attrid:(t2PWeightId)
357                 SET outgoingFlowRelationsWeight:(val)
358                 # Get the Ids of the EndPoints of the FlowRelations
359                 CC "Core" GET_CONNECTOR_ENDPOINTS objid:(VAL outgoingFlowRelationsId)
360
361                 # Get the Number Of Tokens of the Place
362                 CC "Core" GET_ATTR_VAL objid:(toobjid) attrid:(numberOfTokenId)
363                 SET placeNumberOfToken:(val)
364                 # Add the newly produced Token
365                 CC "Core" SET_ATTR_VAL objid:(toobjid) attrid:(numberOfTokenId) val:(placeNumberOfToken + outgoingFlowRelationsWeight)
366
367
368             }
```

Live Demo

For more details visit OMI Website



http://www.openmodels.at/web/petrinets/home

To use Petri Nets Simulation Tool download "Remote Access Use" from website of OMI

Further Questions?

tutorial@adoxx.org