The ePNK:
An extensible Petri net tool for PNML

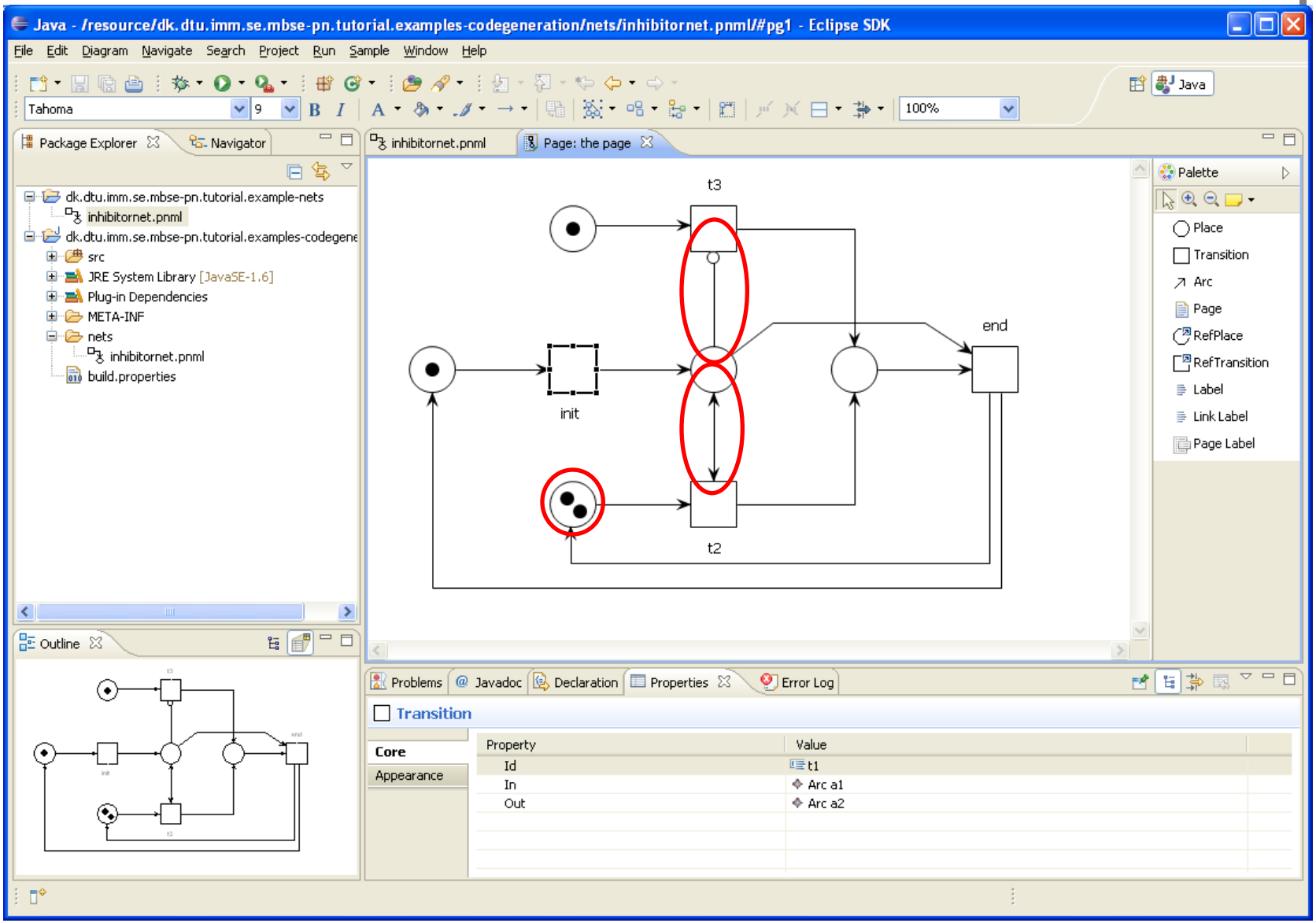
Ekkart Kindler

Denmark's Technical University

DTU Informatics

New Petri net type

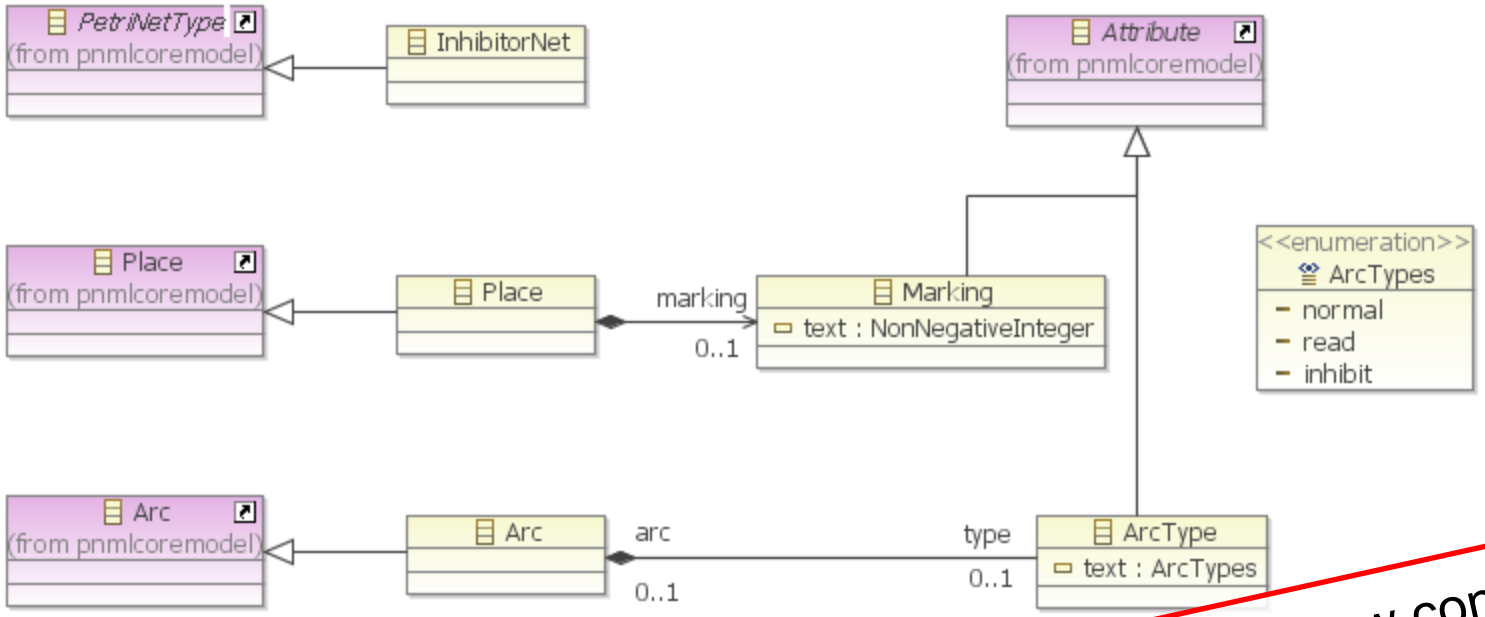
Technical University of Denmark
Informatics and Mathematical Modelling
Ekkart Kindler



Define a Petri Net Type

Technical University of Denmark
Informatics and Mathematical Modelling
Ekkart Kindler

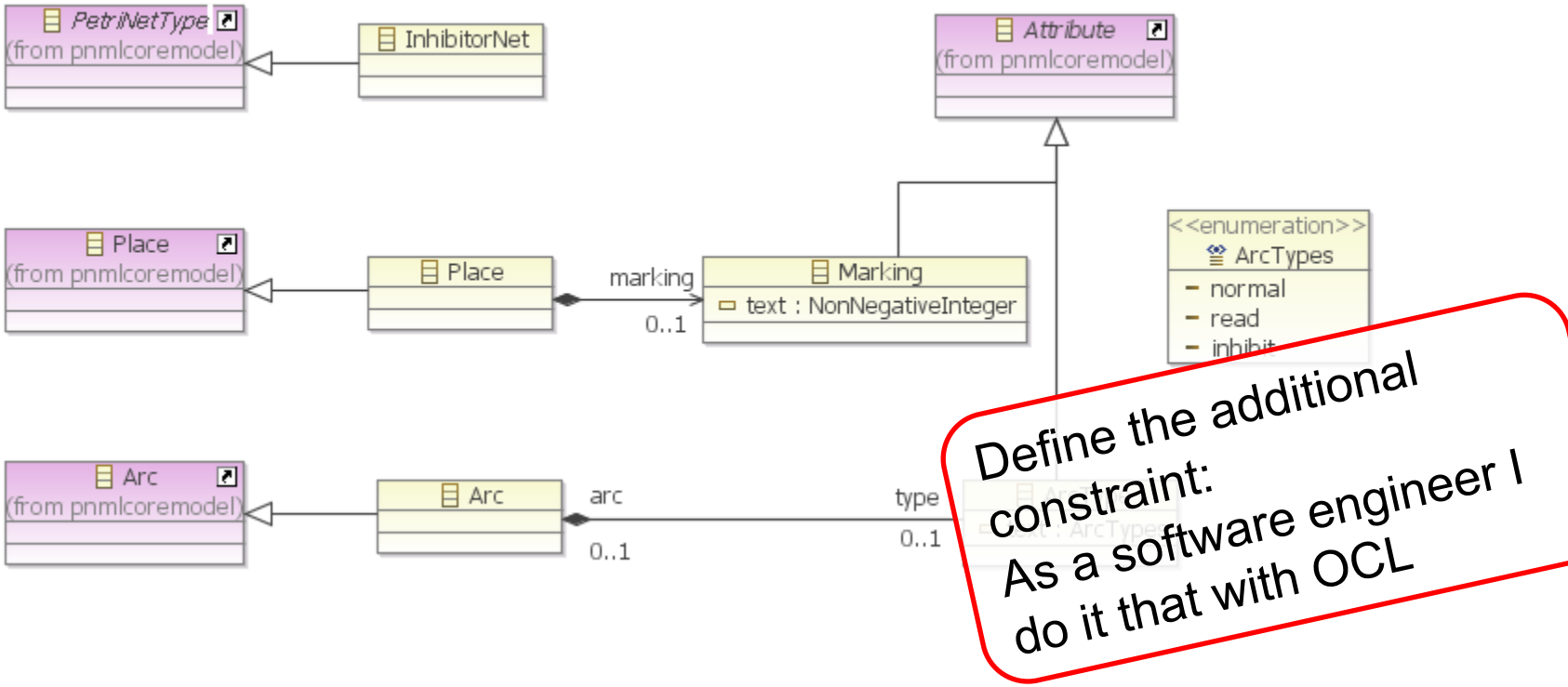
What should it take to define this new Petri Net type conceptually?



Define the new concepts:
As a software engineer, I do it that with a class diagram

Define a Petri Net Type

Technical University of Denmark
Informatics and Mathematical Modelling
Ekkart Kindler

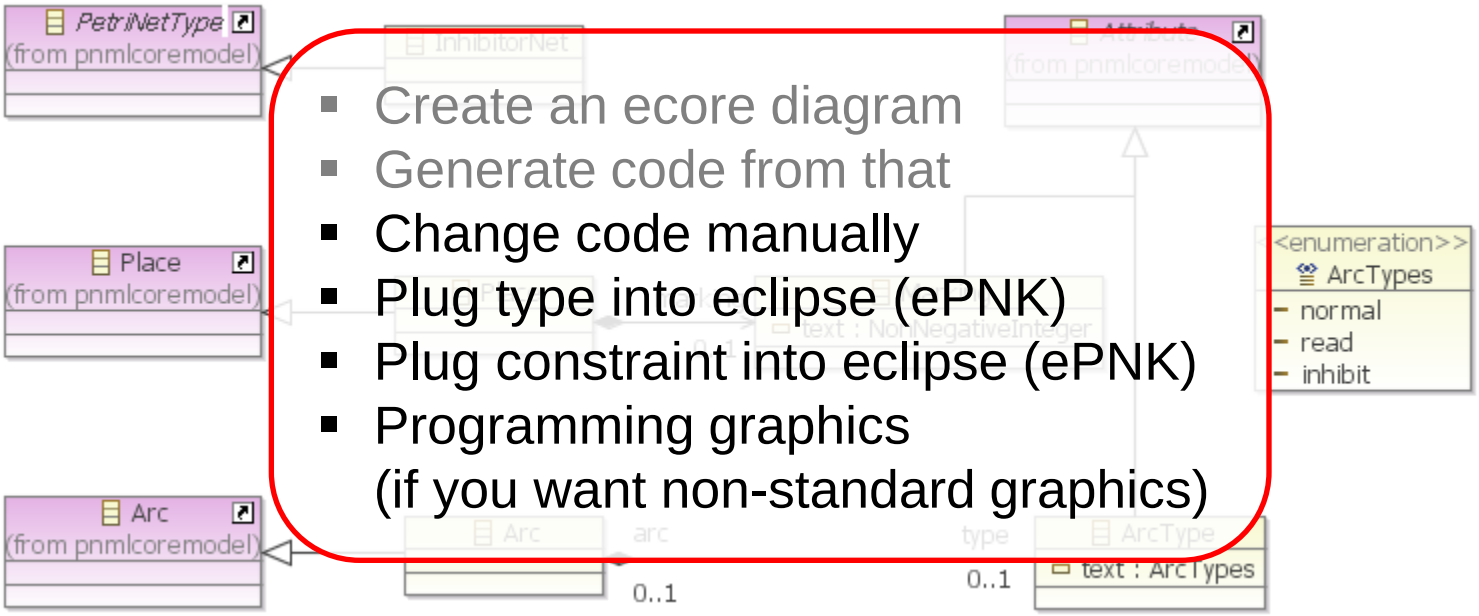



```
( self.source.oclIsKindOf(pnmlcoremodel::PlaceNode) and
  self.target.oclIsKindOf(pnmlcoremodel::TransitionNode) )
or
( self.source.oclIsKindOf(pnmlcoremodel::TransitionNode) and
  self.target.oclIsKindOf(pnmlcoremodel::PlaceNode) and
  not ( self.type.text = ArcTypes::inhibit ) )
```

Define a Petri Net Type

Technical University of Denmark
Informatics and Mathematical Modelling
Ekkart Kindler

What does it take to implement it now?



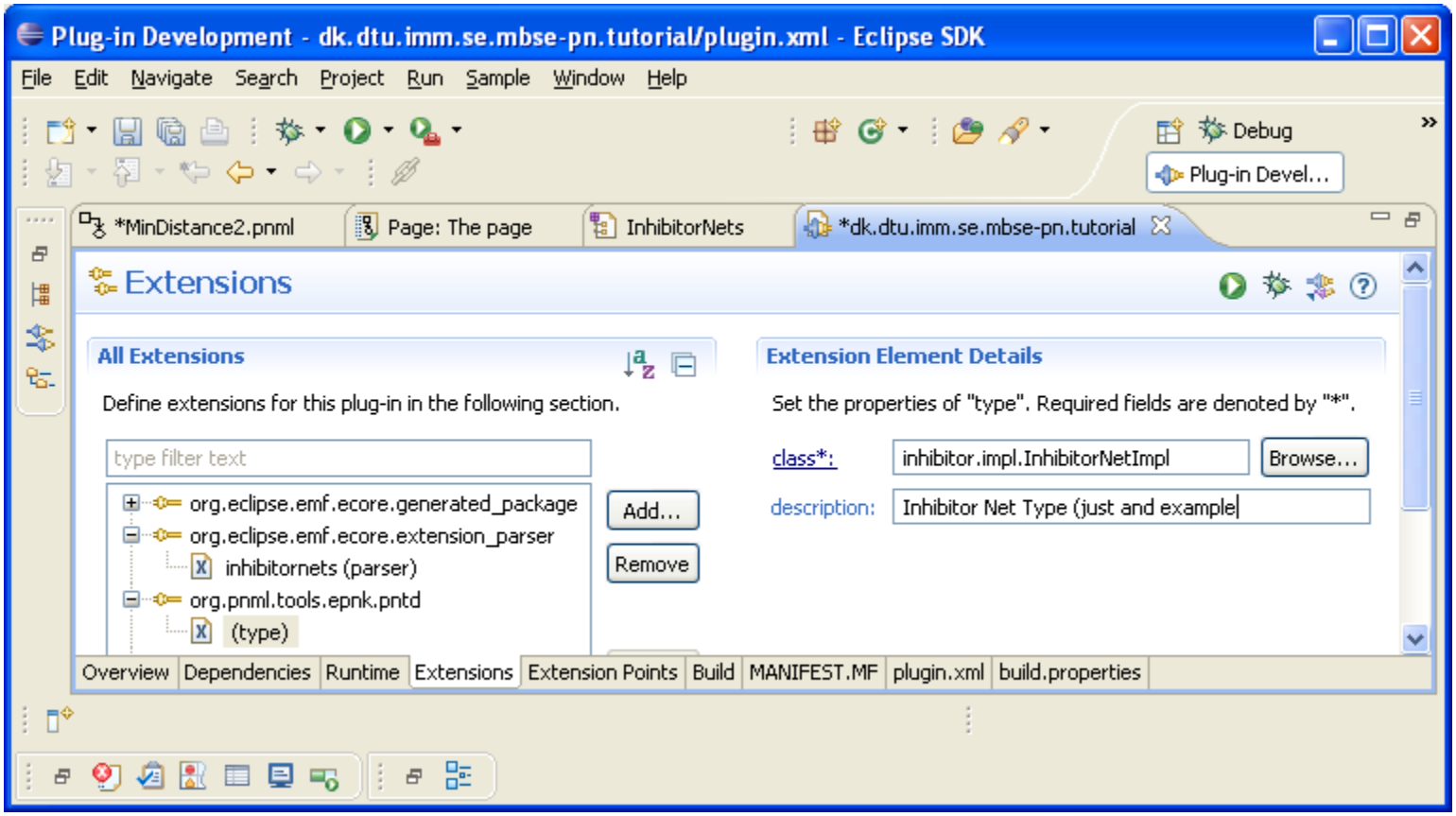

```
( self.source.oclIsKindOf(pnmlcoremodel::PlaceNode) and
  self.target.oclIsKindOf(pnmlcoremodel::TransitionNode) )
or
( self.source.oclIsKindOf(pnmlcoremodel::TransitionNode) and
  self.target.oclIsKindOf(pnmlcoremodel::PlaceNode) and
  not ( self.type.text = ArcTypes::inhibit ) )
```

Change code manually


```
package inhibitornets.impl;

[…]

public class InhibitorNetImpl extends PetriNetTypeImpl implements
   InhibitorNet {

   public InhibitorNetImpl() {
        super();
   }

   protected EClass eStaticClass() {
        return InhibitornetsPackage.Literals.INHIBITOR_NET;
   }

   public String toString() {
        return "http://dk.dtu.imm.se.mbse-pn.tutorial.inhibitornet";

   }

}
```

Making constructor of generated code public is not so nice! Extend the class and make the constructor of the extended class public.

Plug in net type



Plug in constraint


```
<extension
    point="org.eclipse.emf.validation.constraintProviders">

      [about 40 other boring but technically important lines]


  <![CDATA[
  ( self.source.oclIsKindOf(pnmlcoremodel::PlaceNode) and
    self.target.oclIsKindOf(pnmlcoremodel::TransitionNode) )
  or
  ( self.source.oclIsKindOf(pnmlcoremodel::TransitionNode) and
    self.target.oclIsKindOf(pnmlcoremodel::PlaceNode) and
    not ( self.type.text = ArcTypes::inhibit ) )
  ]]>
     </constraint>
    </constraints>
  </constraintProvider>
</extension>
```

Unfortunately, this is a bit technical

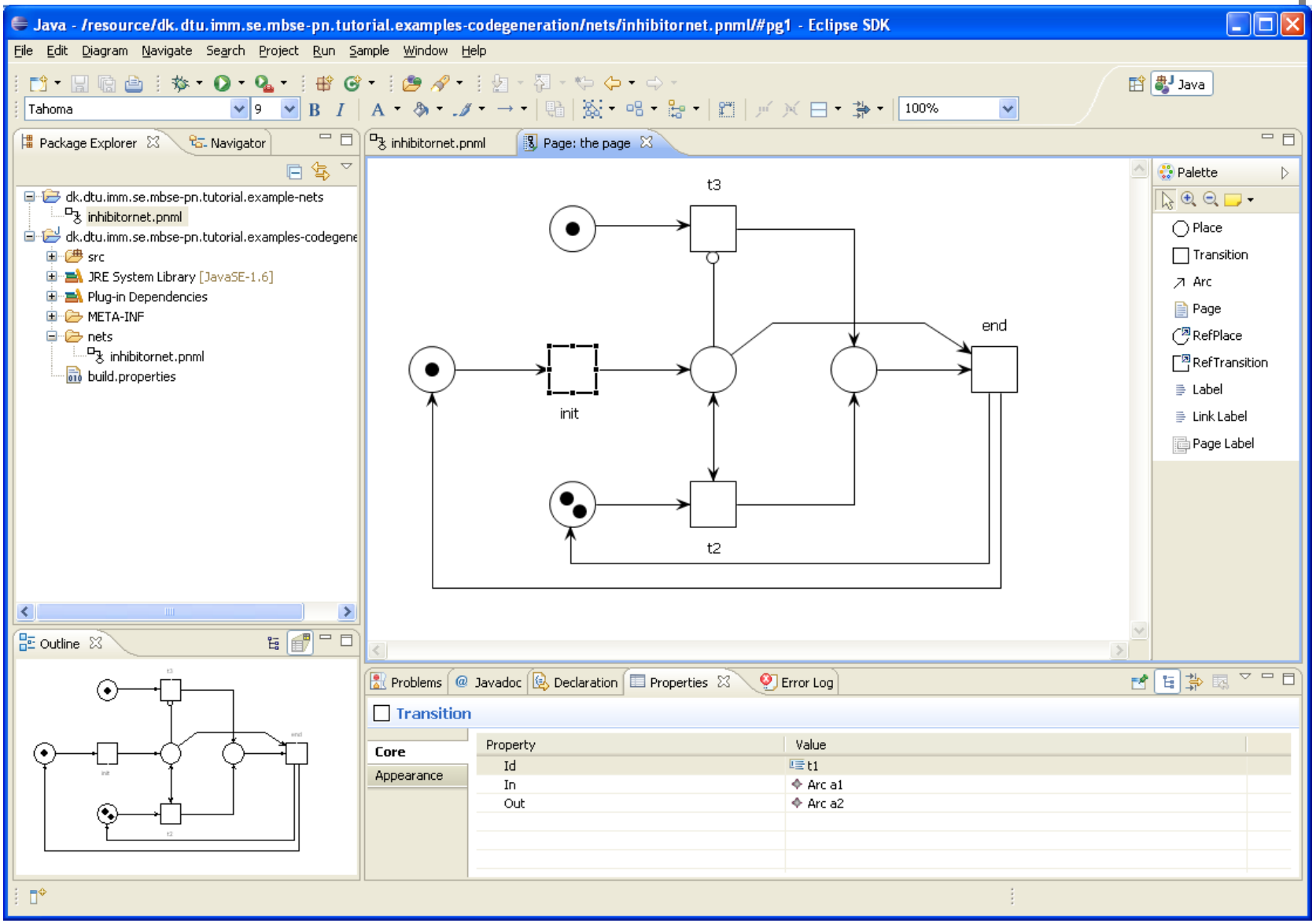
Constraints can also be programmed

Programm Graphics


```
public class InhibitornetsArcFigure extends ArcFigure {
[...]
  private void setGraphics() {
    RotatableDecoration targetDecorator = null;
    RotatableDecoration sourceDecorator = null;

    if (type.equals(ArcTypes.NORMAL)) {
      targetDecorator = new ReisigsArrowHeadDecoration();

    } else if (type.equals(ArcTypes.INHIBIT)) {
      CircleDecoration circleDecoration =
        new CircleDecoration();
      circleDecoration.setLineWidth(this.getLineWidth());
      targetDecorator = circleDecoration;

    } else if (type.equals(ArcTypes.READ)) {
      targetDecorator = new ReisigsArrowHeadDecoration();
      sourceDecorator = new ReisigsArrowHeadDecoration();
[...]
```

Just a glimpse!

Result


Java - /resource/dk.dtu.imm.se.mbse-pn.tutorial.examples-codegeneration/nets/inhibitornet.pnml/#pg1 - Eclipse SDK
File Edit Diagram Navigate Search Project Run Sample Window Help
Package Explorer
Navigator
dk.dtu.imm.se.mbse-pn.tutorial.example-nets
inhibitornet.pnml
dk.dtu.imm.se.mbse-pn.tutorial.examples-codegene
src
JRE System Library [JavaSE-1.6]
Plug-in Dependencies
META-INF
nets
inhibitornet.pnml
build.properties
inhibitornet.pnml
Page: the page
t3
end
init
t2
Palette
Place
Transition
Arc
Page
RefPlace
RefTransition
Label
Link Label
Page Label
Outline
Problems
Javadoc
Declaration
Properties
Error Log
Transition
Core
Appearance
Property
Value
Id
t1
In
Arc a1
Out
Arc a2

Technical details

You can look up some details in the version of the ePNK that was deployed in the SE2 course:

`org.pnml.tools.epnk.extensions.tutorial.types`

`org.pnml.tools.epnk.extensions.tutorial.types.edit`

In order to see these projects to your workspace:

- Open “Plug-ins” view
- Select the resp. plug-in project(s)
- Right-click and choose Import As → Source Project