

# Tools and Methodologies for GMF



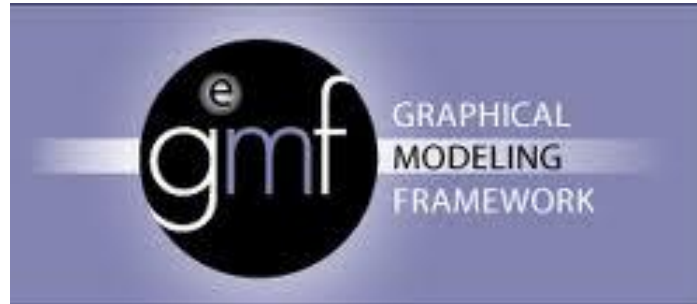
Aurélien Pupier  
R&D Engineer, Studio Project Leader  
GMF-Runtime Committer  
@apupier

# Agenda

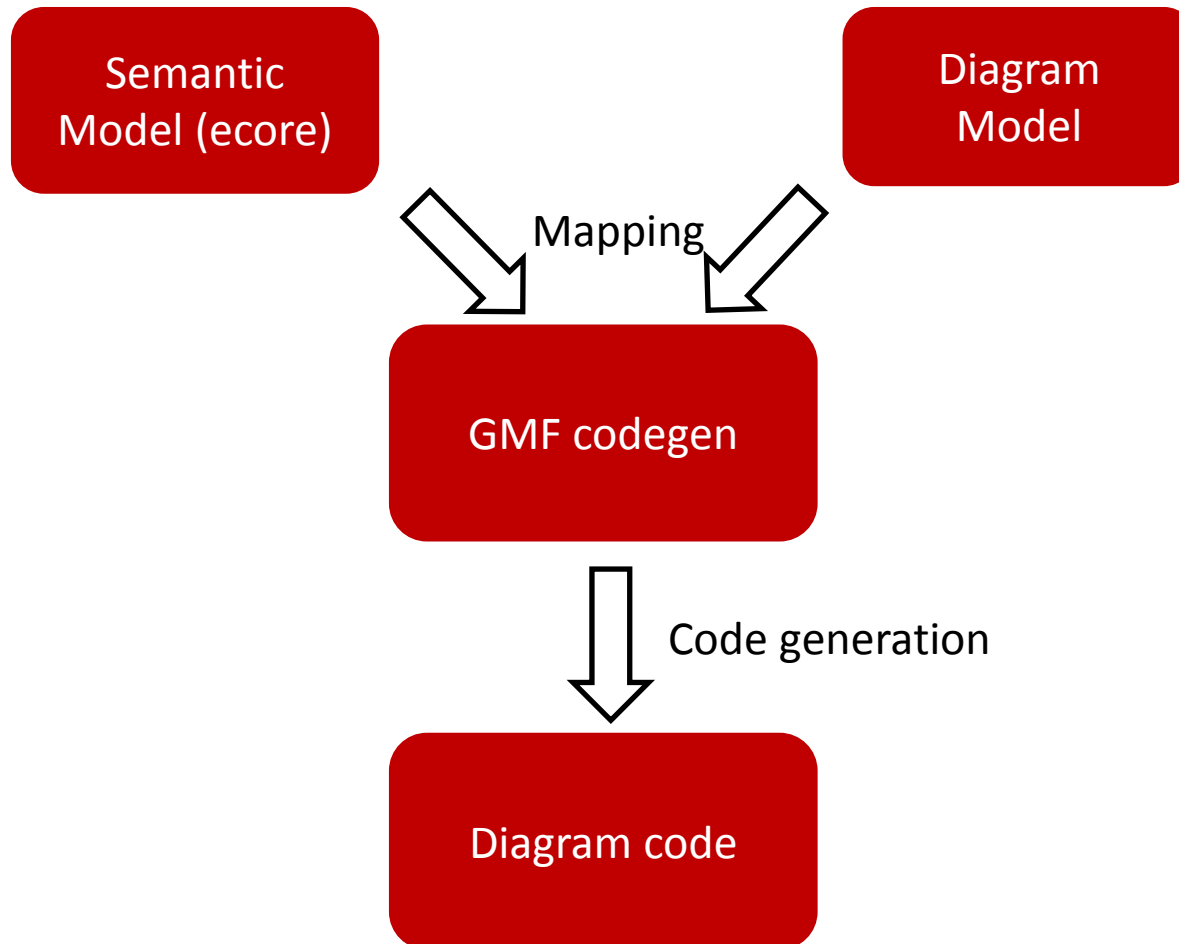
- ↳ Show me a GMF editor
- ↳ GMF Big Picture
- ↳ Methodologies
- ↳ Tooling

# GMF BIG PICTURE

# GMF



# GMF Basic Workflow



# What happens after the first generation?

- ↳ Requirements update
- ↳ Introduce specific behavior
- ↳ I'm not able to do a one shot!

# METHODOLOGIES

# Update the model!

- ↳ Straightforward
- ↳ Best maintainability
- ↳ Easiest solution

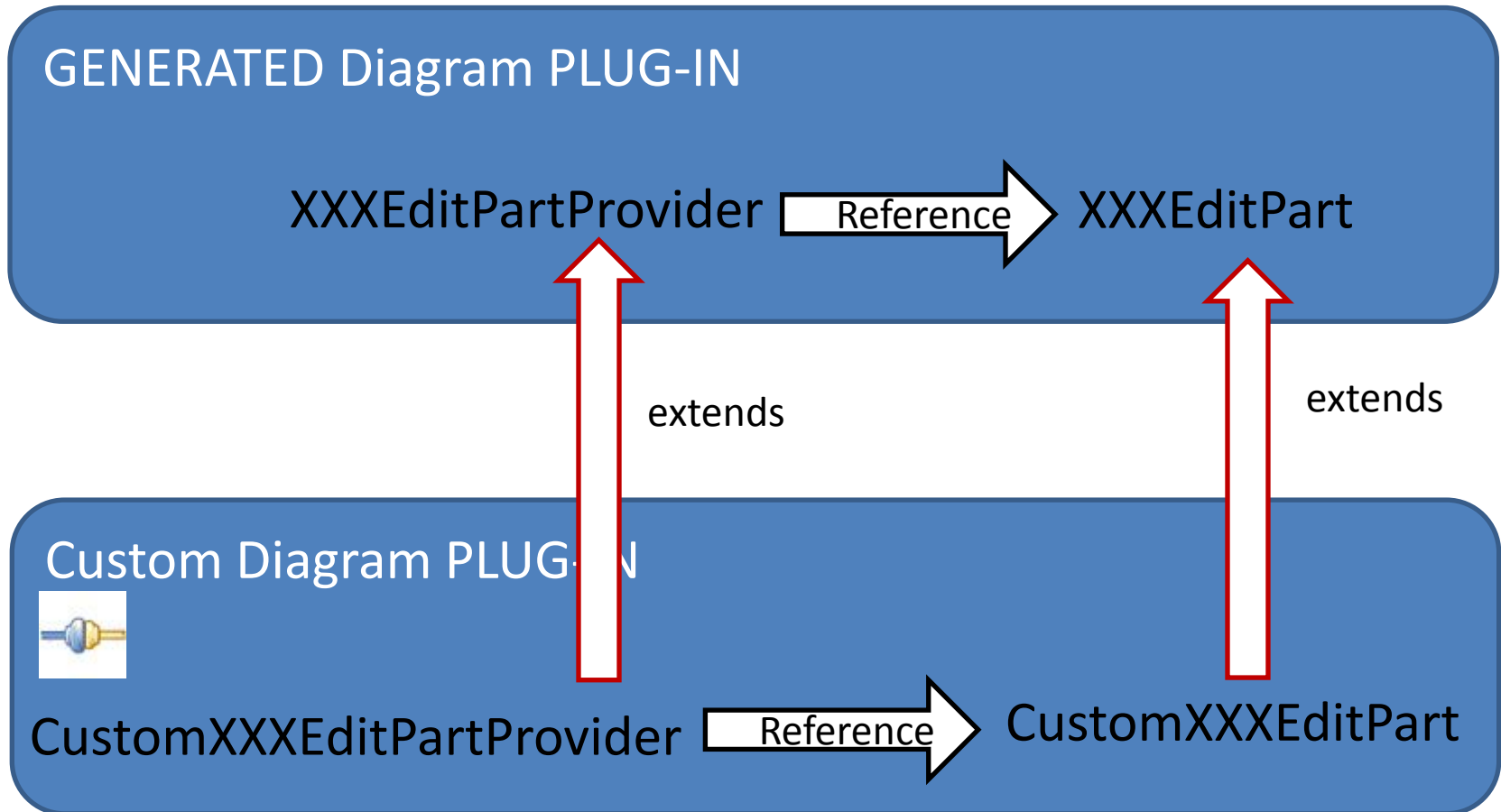


# GMF Extension points

↳ 24

↳ Classical Eclipse way

# Extension points – Typical pattern



# Generation templates

## ↳ When?

- ↳ Apply to a whole kind of elements
- ↳ Ultimately

## ↳ Useful for:

- ↳ Regeneration
- ↳ New element introduced

# Generation aspect templates

- ↳ Reduce amount of code to maintain
- ↳ Avoid code duplication
- ↳ *Down grain from class to method*

# Generation aspect templates

```
«AROUND handleNotificationEventBody FOR gmfgen::GenExternalNodeLabel->
    Object feature = event.getFeature();
    «EXPAND xpt::diagram::editparts::Common::handleText FOR self->
    //if 'to' link changed
    if(event.getEventType() == org.eclipse.emf.common.notify.Notification.SET
        && feature.equals(org.bonitasoft.studio.model.process.ProcessPackage.Literals.THROW_LINK_EVENT_TO)){
        refreshLabel();
    }
    super.handleNotificationEvent(event);
«ENDAROUND»
```

```
«IMPORT 'http://www.eclipse.org/gmf/2009/GenModel'»
«DEFINE initializer FOR gmfgen::GenExternalNodeLabel->
    «EXPAND xpt::Common::generatedMemberComment»
    static {
        registerSnapBackPosition(«EXPAND xpt::editor::VisualIDRegistry::typeMethodCall», new org.eclipse.draw2d.geometry.Point(0, 0));
    }
«ENDEDEFINE»
«DEFINE _constructor FOR gmfgen::GenExternalNodeLabel->
    «EXPAND xpt::Common::generatedMemberComment»
    public «editPartClassName»(org.eclipse.gmf.runtime.notation.View view) {
        super(view);
    }
«ENDEDEFINE»
«DEFINE createDefaultEditPoliciesBody FOR gmfgen::GenExternalNodeLabel->
    super.createDefaultEditPolicies();
    installEditPolicy(org.eclipse.gef.EditPolicy.DIRECT_EDIT_ROLE, new org.eclipse.gmf.runtime.diagram.ui.editpolicies.LabelDirectEditPolicy());
    installEditPolicy(org.eclipse.gef.EditPolicy.SELECTION_FEEDBACK_ROLE, new «getDiagram().getTextSelectionEditPolicyQualifiedClassName»());
    «EXPAND xpt::diagram::editparts::Common::behaviour FOR self->
«ENDEDEFINE»
«DEFINE getBorderItemLocator FOR gmfgen::GenExternalNodeLabel->
    «EXPAND xpt::Common::generatedMemberComment»
    public org.eclipse.gmf.runtime.diagram.ui.figures.IBorderItemLocator getBorderItemLocator() {
        org.eclipse.draw2d.IFigure parentFigure = getFigure().getParent();
        if (parentFigure != null && parentFigure.getLayoutManager() != null) {
            Object constraint = parentFigure.getLayoutManager().getConstraint(getFigure());
            return (org.eclipse.gmf.runtime.diagram.ui.figures.IBorderItemLocator) constraint;
        }
        return null;
    }
«ENDEDEFINE»
«DEFINE refreshBounds FOR gmfgen::GenExternalNodeLabel->
    «EXPAND xpt::Common::generatedMemberComment»
    public void refreshBounds() {
```

Another tip to reduce size of  
custom generation templates?

# Utility plugin – why?

```
«AROUND extendsList FOR gmfgen::GenExternalNodeLabel»extends org.bonitasoft.studio.common.gmf.CustomEventLabelEditPart«ENDAROUND»
```

```
package org.bonitasoft.studio.common.gmf;

import org.eclipse.draw2d.IFigure;
import org.eclipse.draw2d.geometry.Dimension;
import org.eclipse.draw2d.geometry.Point;
import org.eclipse.gef.GraphicalEditPart;
import org.eclipse.gmf.runtime.diagram.ui.editparts.LabelEditPart;
import org.eclipse.gmf.runtime.diagram.ui.figures.LabelLocator;
import org.eclipse.gmf.runtime.notation.NotationPackage;
import org.eclipse.gmf.runtime.notation.View;

/**
 * @author Baptiste Mesta
 */
public class CustomEventLabelEditPart extends LabelEditPart {

    /**
     * @param view
     */
    public CustomEventLabelEditPart(View view) {
        super(view);
    }

    public void refreshBounds() {
        IFigure refFigure = ((GraphicalEditPart) getParent()).getFigure();
        int dx = ((Integer) getStructuralFeatureValue(NotationPackage.eINSTANCE
            .getLocation_X())).intValue();
        int dy = ((Integer) getStructuralFeatureValue(NotationPackage.eINSTANCE
            .getLocation_Y())).intValue();
        Point offset = new Point(dx, dy);
        if(getFigure().getParent() != null){
            getFigure().getParent().setConstraint(getFigure(),
                new LabelLocator(refFigure, offset, getKeyPoint()) {

                public void relocate(IFigure target) {
                    Point location = getReferencePoint().getTranslated(
                        getOffset());
                    location.translate(-target.getBounds().width / 2, 0);
                    target.setLocation(location);
                    target.setSize(new Dimension(
                        target.getPreferredSize().width, target
                            .getPreferredSize().height));
                }

                protected Point getReferencePoint() {
                    return getLabelLocation(parent);
                }
            });
        }
    }

    @Override
    public Point getReferencePoint() {
        return getLabelLocation(((GraphicalEditPart) getParent()).getFigure());
    }
}
```

# Utility plugin – how?

- platform:/resource/org.bonitasoft.studio-models/process.gmfgen
  - Gen Editor Generator org.bonitasoft.studio.model.process.diagram
    - Gen Diagram MainProcessEditPart
      - Gen Plugin Bonita process editor**
      - Gen Editor View org.bonitasoft.studio.model.process.diagram.part
      - Gen Navigator ProcessNavigatorContentProvider
      - Gen Diagram Updater ProcessDiagramUpdater

Selection | Parent | List | Tree | Table | Tree with Columns

Tasks | Problems | History | Console | Search | Debug | Progress | Properties | JUnit | Cross References | Builds

Property	Value
Activator Class Name	ProcessDiagramEditorPlugin
ID	org.bonitasoft.studio.diagram
Name	Bonita process editor
Printing Enabled	false
Provider	BonitaSoft S.A.
Required Plugin Identifiers	org.eclipse.draw2d, org.eclipse.gmf.runtime.draw2d.ui, org.eclipse.gmf.runtime.lite.svg, org.bonitasoft.studio.common,
Version	1.0.0.qualifier








# Modify Generated code

~~@Generated NOT~~

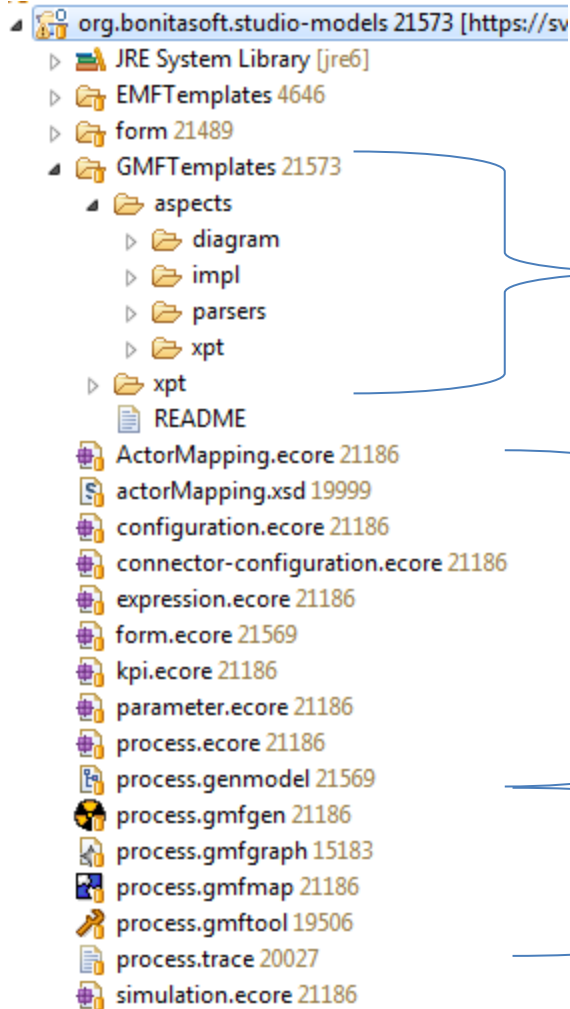
## Sandbox

# Projects Structure (1/2)

## GMF-Methodo (9)

- ▶  org.bonitasoft.studio-models 21573 [<https://svn.bonitasoft.com/org.bonitasoft.studio-models>]
- ▶  org.bonitasoft.studio.diagram
- ▶  org.bonitasoft.studio.diagram.custom 21573 [<https://svn.bonitasoft.com/org.bonitasoft.studio.diagram.custom>]
- ▶  org.bonitasoft.studio.diagram.form
- ▶  org.bonitasoft.studio.diagram.form.custom 21528 [<https://svn.bonitasoft.com/org.bonitasoft.studio.diagram.form.custom>]

# Projects Structure (2/2)

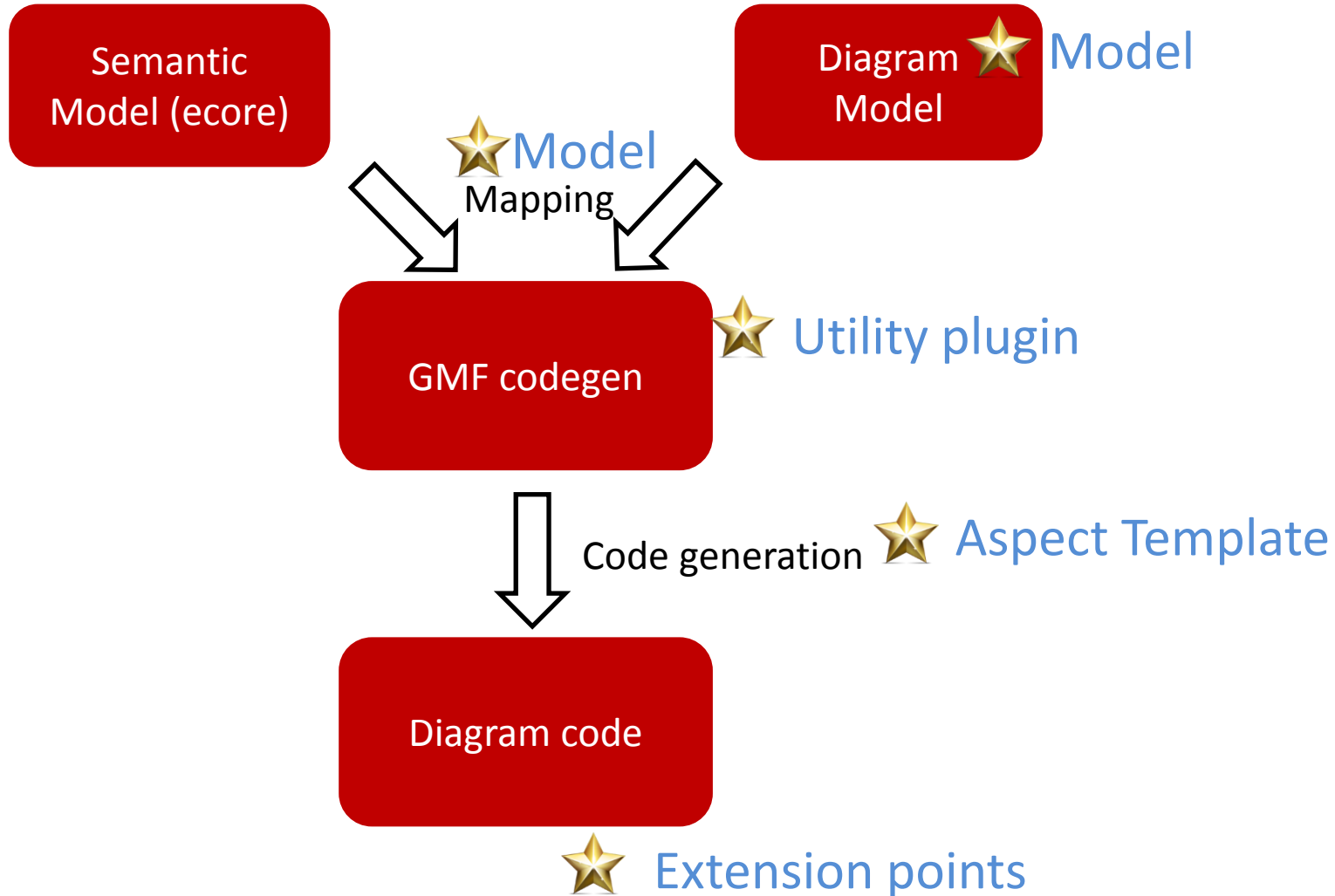


Generation Templates

Semantic Model

Diagram Model

# GMF Workflow

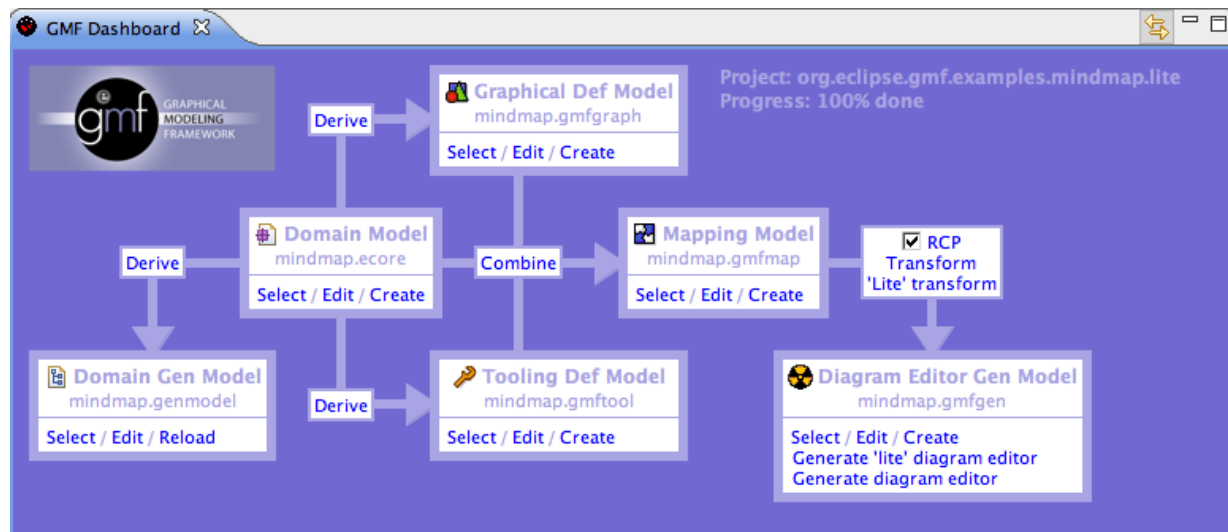


# TOOLING

# GMF-Tooling

## Official tooling of GMF

- Several editors
- Wizards
- GMF Dashboard



# SWTBot4GEF

- ↳ Automated UI tests

- ↳ Simple API:

  - ↳ click

  - ↳ drag

  - ↳ select

  - ↳ activateTool

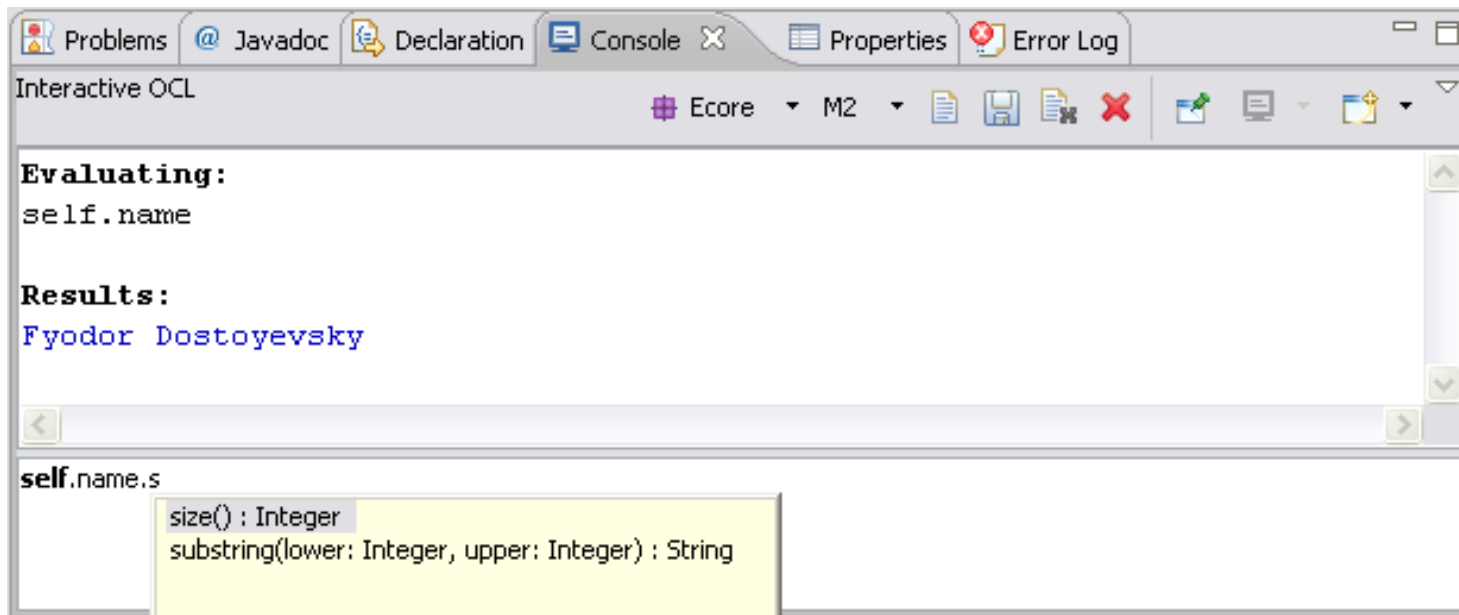
# Code generation Ant Task

- ↳ Generate code at build time
- ↳ Avoid *reconcilier* issues
- ↳ Avoid modification of your *sandbox* committed



# OCL Editor and Interpreter

## Write and Test constraints



# Simple Mapping Editor (1/2)

The screenshot displays the Simple Mapping Editor interface. The main workspace shows a diagram with two nodes: `companies:Company` and `departments:Depar...`. The `companies:Company` node has sub-nodes `Managers` (with `managers:Manager`) and `Departments` (with `departments:Department`). The `departments:Depar...` node has a sub-node `employees:Employee`. A dashed box surrounds the `employees:Employee` node, and an arrow labeled `<<parent>>` points from it to the `companies:Company` node. Below these nodes are two ovals labeled `Client` and `Partner`.

The right-hand side features a **Palette** with the following items:

- TopNode
- Compartment
- LabelNode
- Sub Node
- ParentNode
- Link Mapping

The bottom section shows the **SimpleLinkMapping** properties table:

Category	Property	Value
Core	Property	
Appearance	Domain meta information	
	Containment Feature	
Connection	Element	
	Source Feature	
Label	Target Feature	<code>partners : Company</code>
	Misc	
Foreground	Related Diagrams	

# Simple Mapping Editor (2/2)

- ↳ All-in-one editor
- ↳ Keep separation of concerns available
- ↳ Eating your own dog food

# EDapt



- ↳ Big underlying model modifications
- ↳ PoC for BOS 6.0

# Keep in mind

- ↳ Great tooling ecosystem
- ↳ Proven methodologies for continuous improvement

# To go further

↳ Eclipse GMF Forum:

<http://www.eclipse.org/forums/index.php/f/16/>

↳ Eclipse GMF Wiki:

<http://wiki.eclipse.org/GMF>

↳ Talk together 😊

↳ [aurelien.pupier@bonitasoft.com](mailto:aurelien.pupier@bonitasoft.com)

Twitter: *@apupier*

Eclipse blog: [www.bonitasoft.org/blog/category/eclipse](http://www.bonitasoft.org/blog/category/eclipse)

Talk to me about Bonita, BPMN2, Eclipse Modeling and more

↳ Company: [www.bonitasoft.com](http://www.bonitasoft.com)

↳ Community: [www.bonitasoft.org](http://www.bonitasoft.org)

↳ Twitter: *@bonitasoft*