Graphical Editors – 2.

GMF

GMF

- Graphical Modeling Framework
- Goal
  - Graphical editing of DSLs
  - Model-based, with few coding (code generation)
  - Uniform framework
  - Quick, incremental feature development
- Developers:
  - Bonitasoft
  - Formerly IBM, Borland

GMF Overview

- Two main components
  - Runtime
    - Framework over EMF and GEF
    - Model and diagram level features
    - Extensible
  - Tooling
    - Model-driven
    - Graphical, tooling and mapping model
    - Target platform: GMF Runtime

GMF Runtime

- Graphical editor framework
  - Re-usable components
  - Standard diagram metamodel (GMF Notation)
    - Separation of domain and diagram metamodel
    - See also OMG Diagram Exchange specification

GMF Runtime

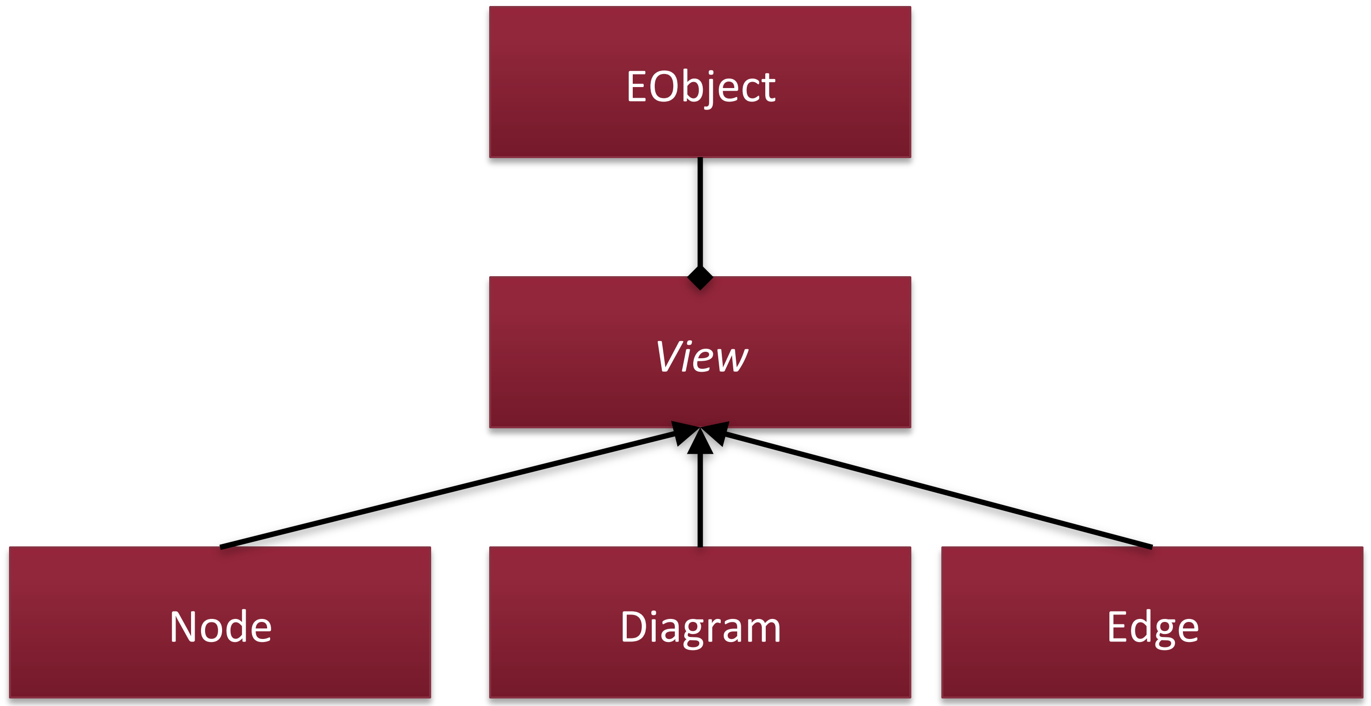
- Command-based framework (over EMF and GEF)
  - Model persistence using EMF runtime
  - Model editing based on GEF
- Further techniques
  - EMF Model Transaction
  - EMF Validation Framework
  - MDT Object Constraint Language (OCL)
  - Apache Batik (SVG)

Notation metamodel

- Presents display information:
  - Color, font, etc.
  - Nodes: position, size, etc.
  - Edges: bendpoints, decoration, etc.
- Notation model referring to domain model
  - Similar to Pictogram + Link model in case of Graphiti
  - Domain model independent, provided by GMF

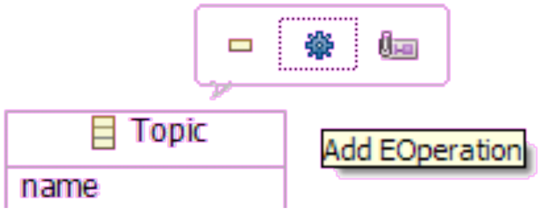
Notation metamodel

- Main element: View
  - Wraps a domain model object
  - get/setElement() for links

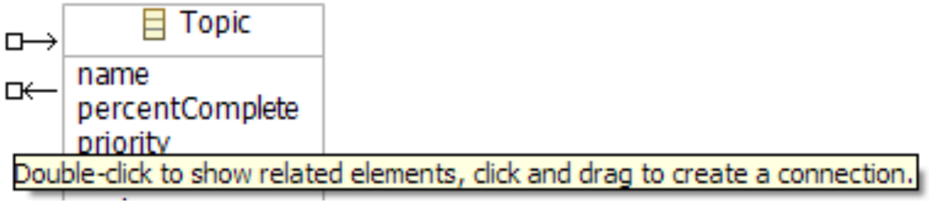


Standard components

Popup Action Bar:

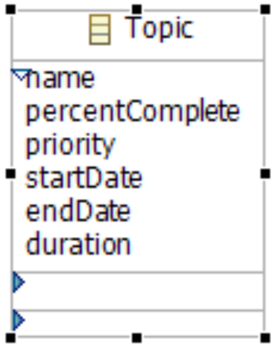


Connection Handle:

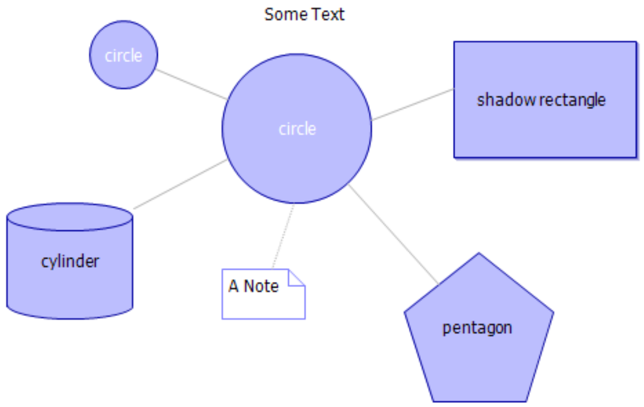


Double-click to show related elements, click and drag to create a connection.

Compartment (collapsible):



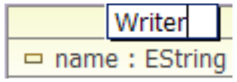
Geometrical Shape:



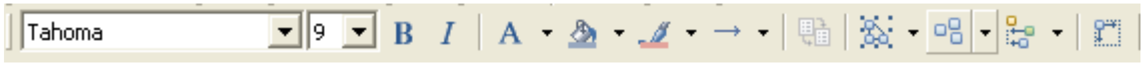
Standard components

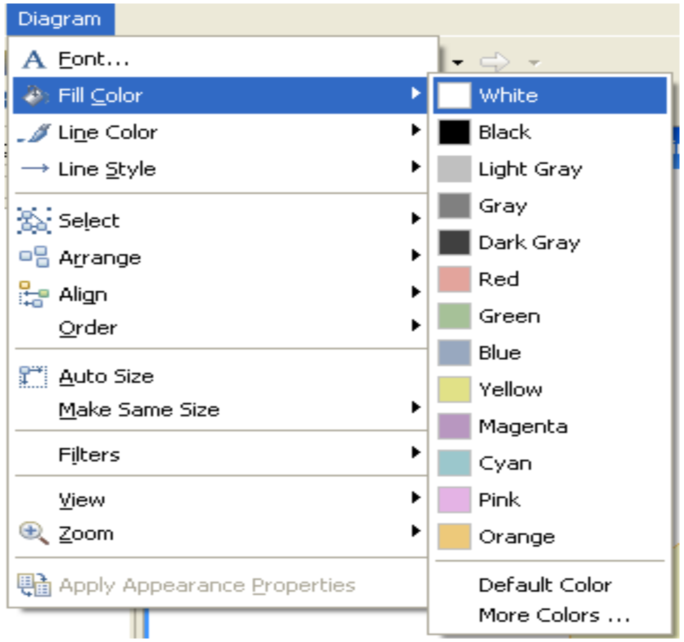
Action:

Direct Edit:

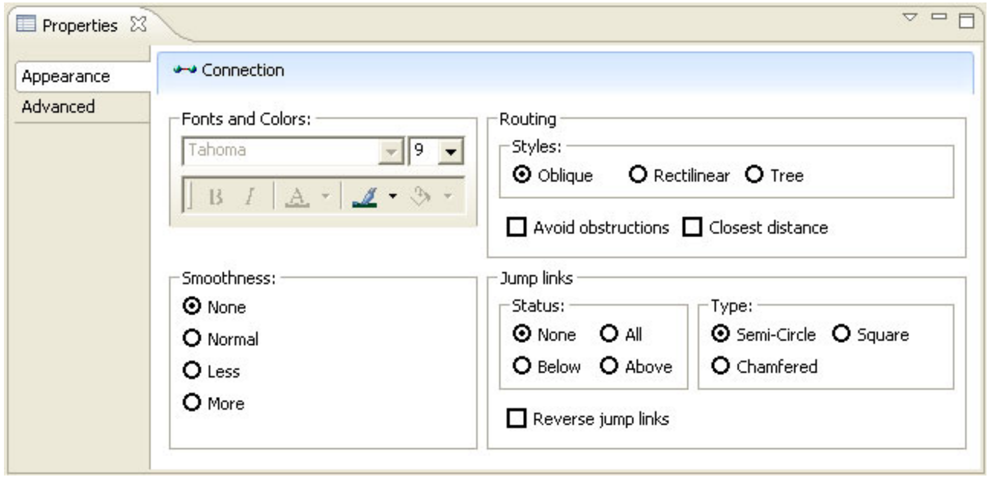


Toolbar:





Properties View:



Fonts and Colors: Tahoma 9

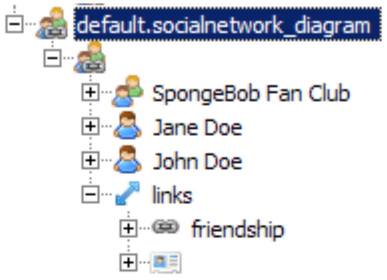
Routing
Styles: [✓ Oblique] Rectilinear Tree
☐ Avoid obstructions ☐ Closest distance

Smoothness: [✓ None] Normal Less More

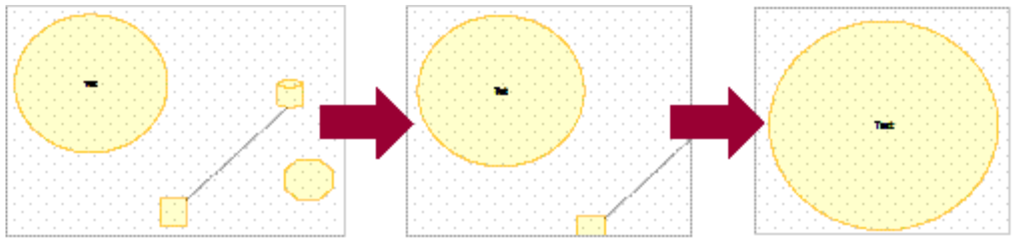
Jump links
Status: [✓ None] All Below Above
Type: [✓ Semi-Circle] Square Chamfered
☐ Reverse jump links

Standard components

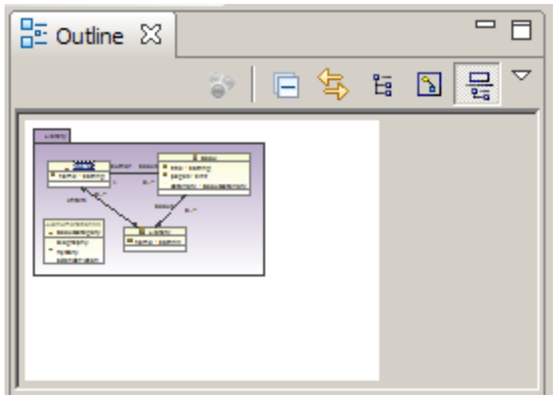
Model navigator



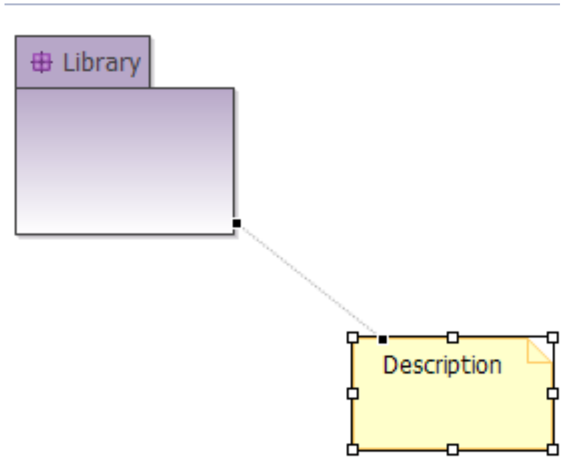
Animated zoom:



Outline view support

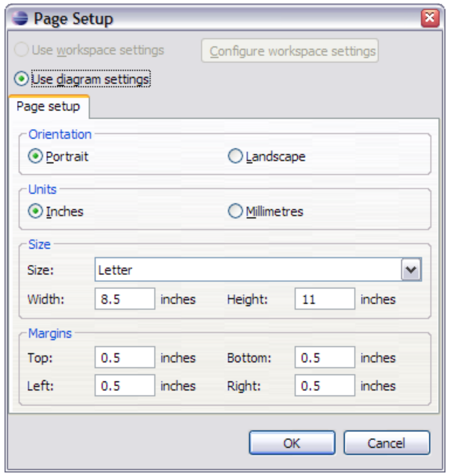


Note Attachment:



Standard components

Print setup:



Page Setup

○ Use workspace settings Configure workspace settings
[✓ Use diagram settings]

Page setup

Orientation
[✓ Portrait] ○ Landscape

Units
[✓ Inches] ○ Millimetres

Size
Size: Letter
Width: 8.5 inches Height: 11 inches

Margins
Top: 0.5 inches Bottom: 0.5 inches
Left: 0.5 inches Right: 0.5 inches

OK Cancel

Print preview:

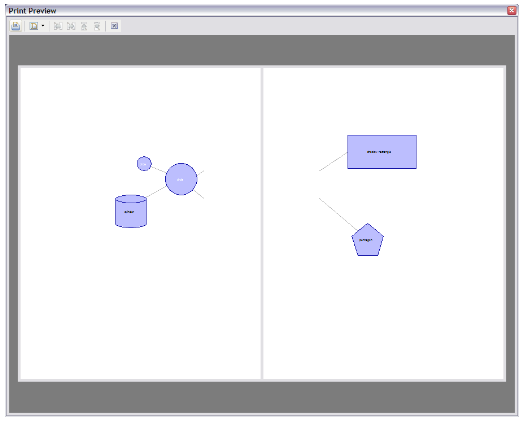
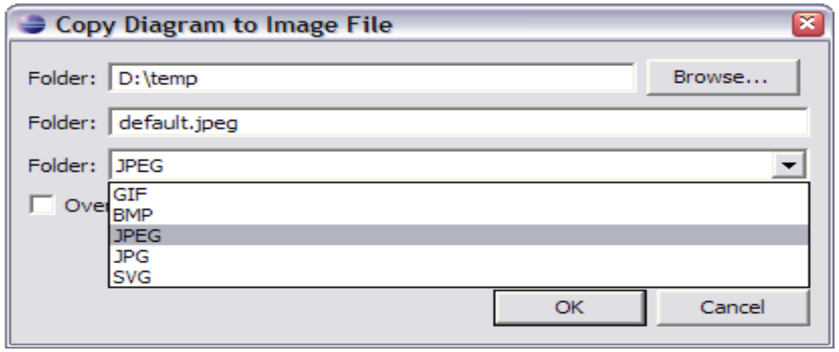


Diagram export to image



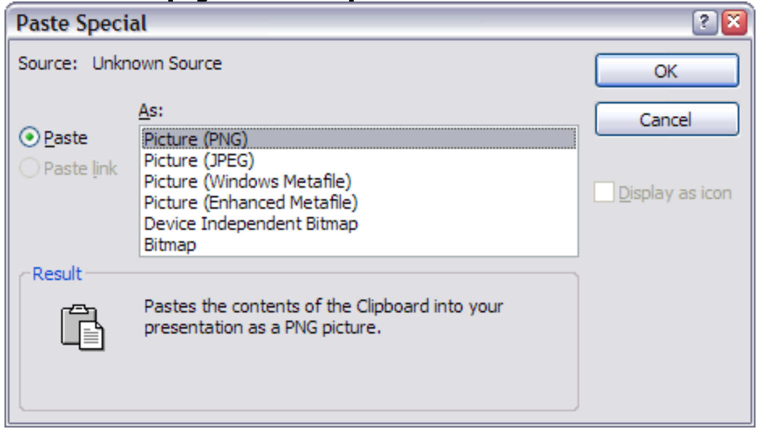
Copy Diagram to Image File

Folder: D:\temp Browse...
Folder: default.jpeg
Folder: JPEG
☐ Ove

GIF
BMP
JPEG
JPG
SVG

OK Cancel

Diagram copy to clipboard



Paste Special

Source: Unknown Source

As:
[✓ Paste] ○ Paste link

Picture (PNG)
Picture (JPEG)
Picture (Windows Metafile)
Picture (Enhanced Metafile)
Device Independent Bitmap
Bitmap

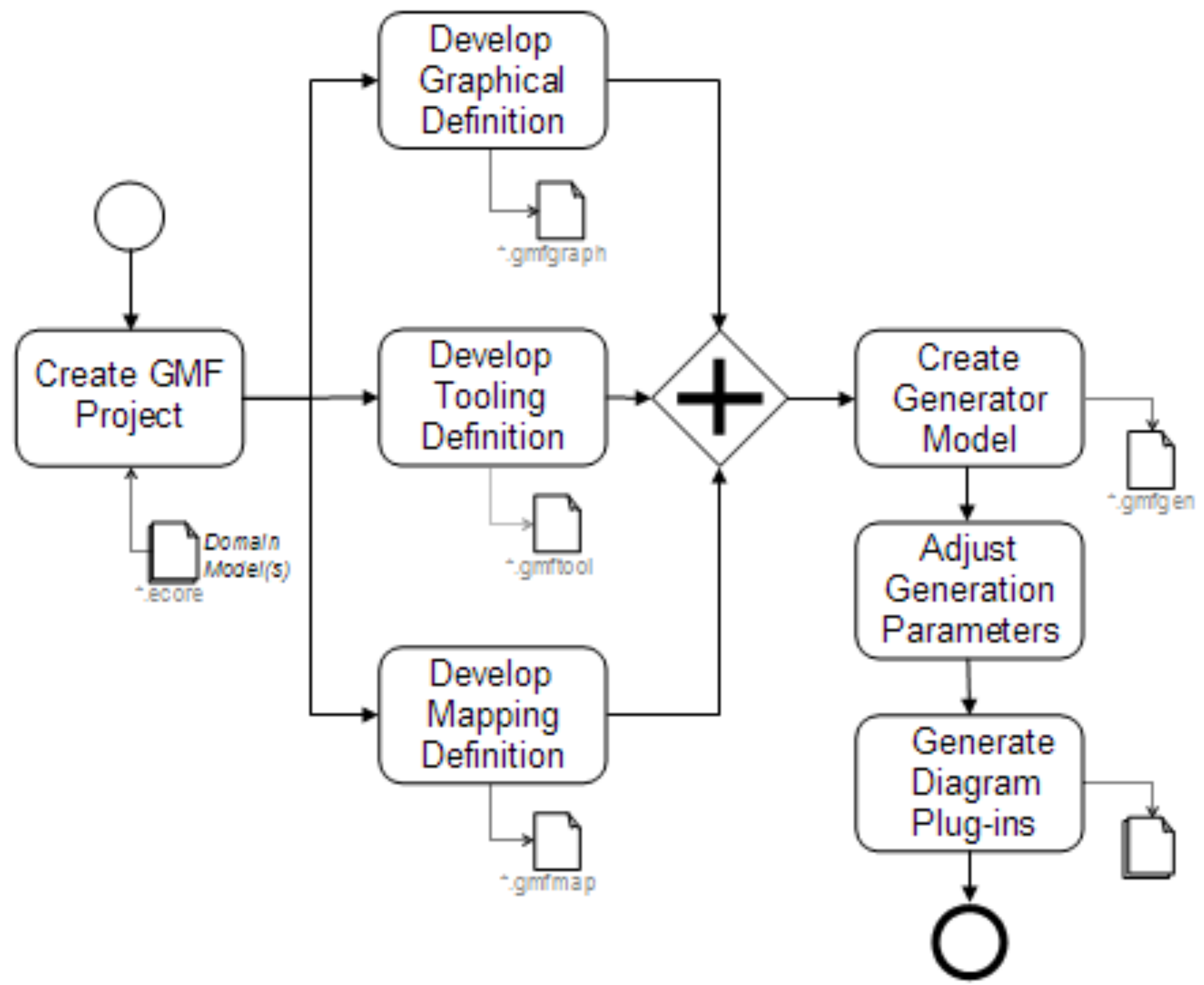
OK Cancel
☐ Display as icon

Result
Pastes the contents of the Clipboard into your presentation as a PNG picture.

GMF Tooling

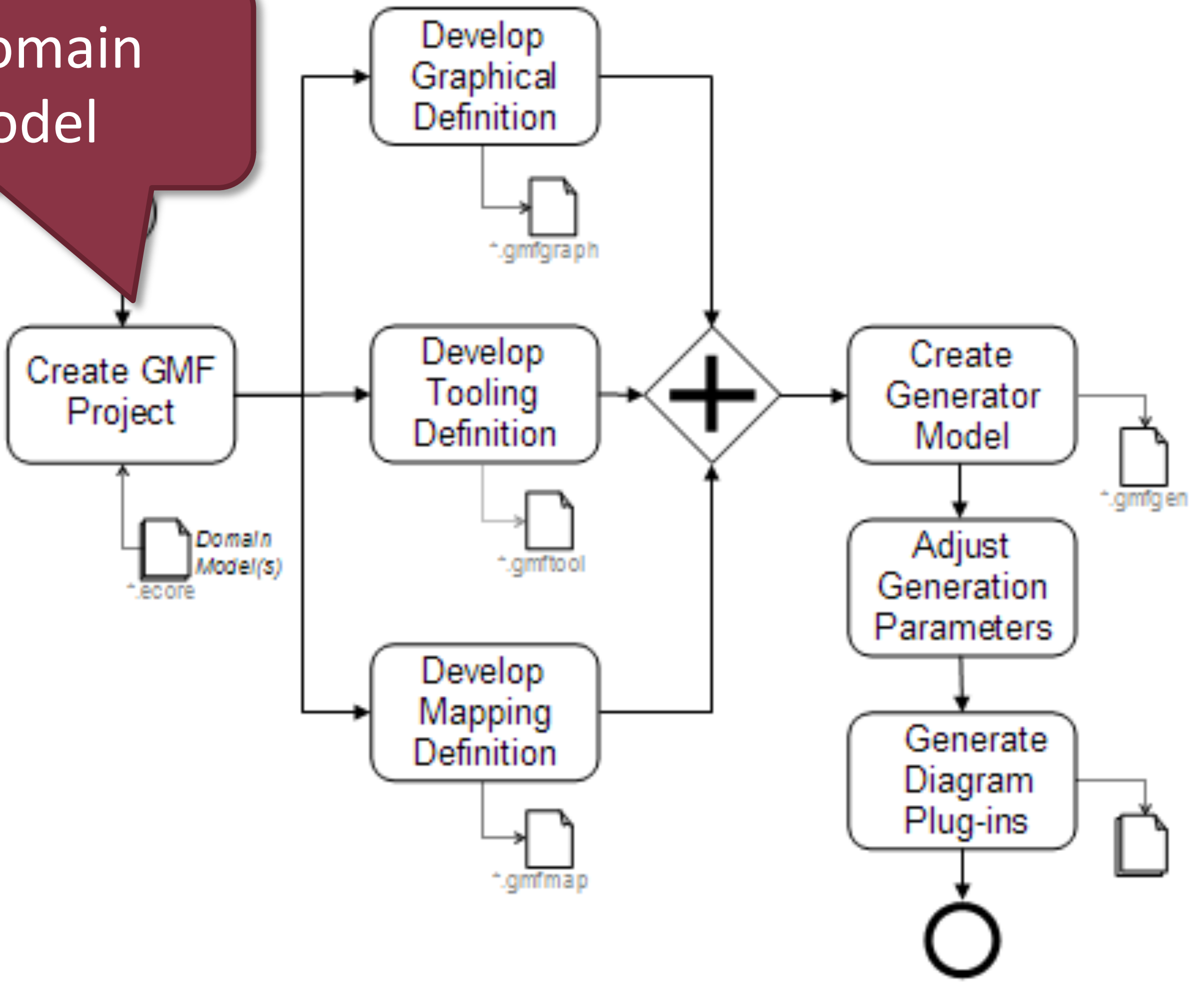
- Goal:
  - Generates model editors
  - Separate diagram and logical models
  - Quick creation of graphical syntaxes
  - Result can be extended

Generating GMF Editors


Create GMF Project
Domain Model(s)
*.ecore
Develop Graphical Definition
*.gmfgraph
Develop Tooling Definition
*.gmftool
Develop Mapping Definition
*.gmfmap
Create Generator Model
*.gmfgen
Adjust Generation Parameters
Generate Diagram Plug-ins

Generating GMF Editors

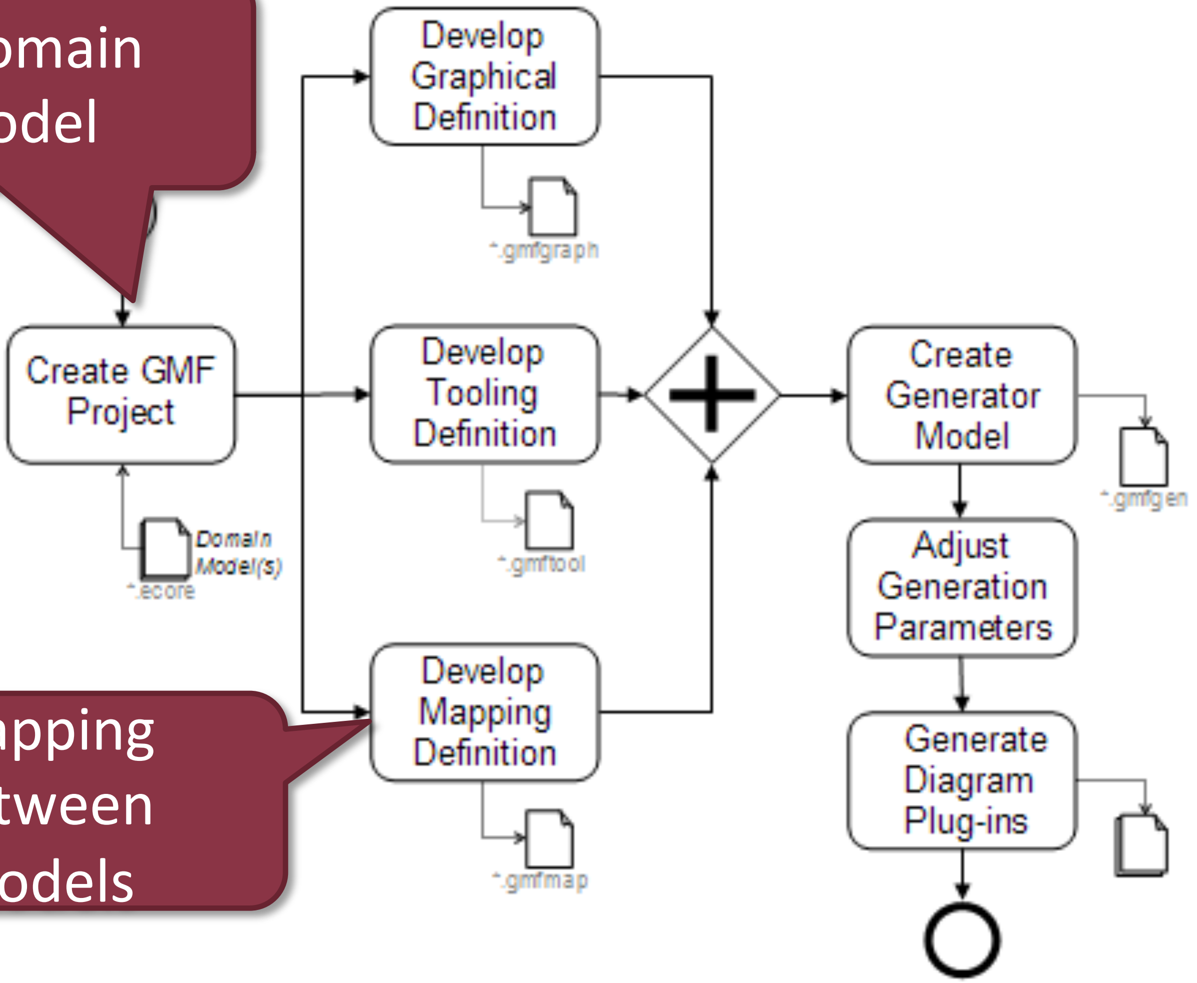
Import domain metamodel


Develop Graphical Definition
*.gmfgraph
Create GMF Project
Domain Model(s)
*.ecore
Develop Tooling Definition
*.gmftool
Develop Mapping Definition
*.gmfmap
Create Generator Model
*.gmfgen
Adjust Generation Parameters
Generate Diagram Plug-ins

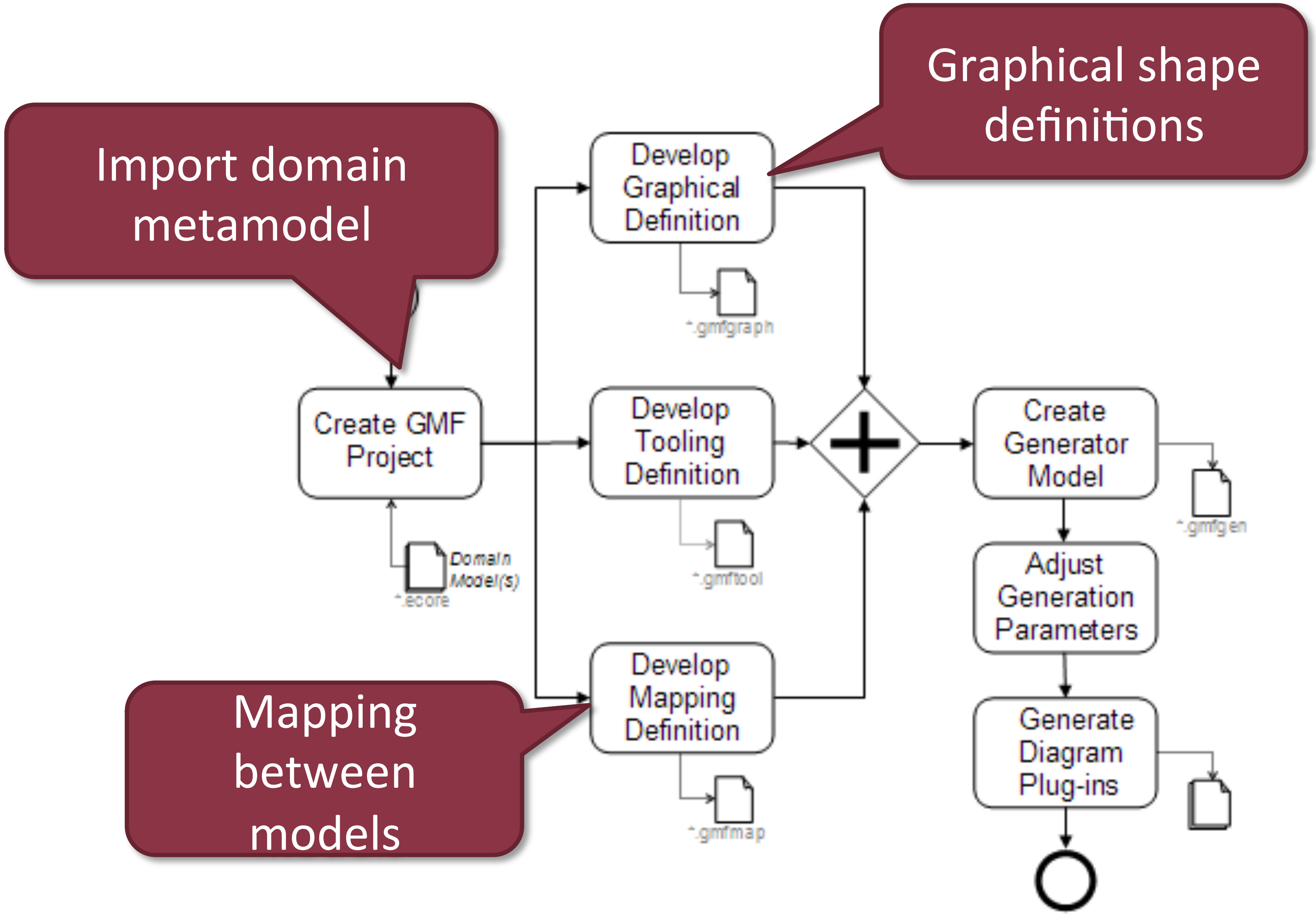
Generating GMF Editors

Import domain metamodel

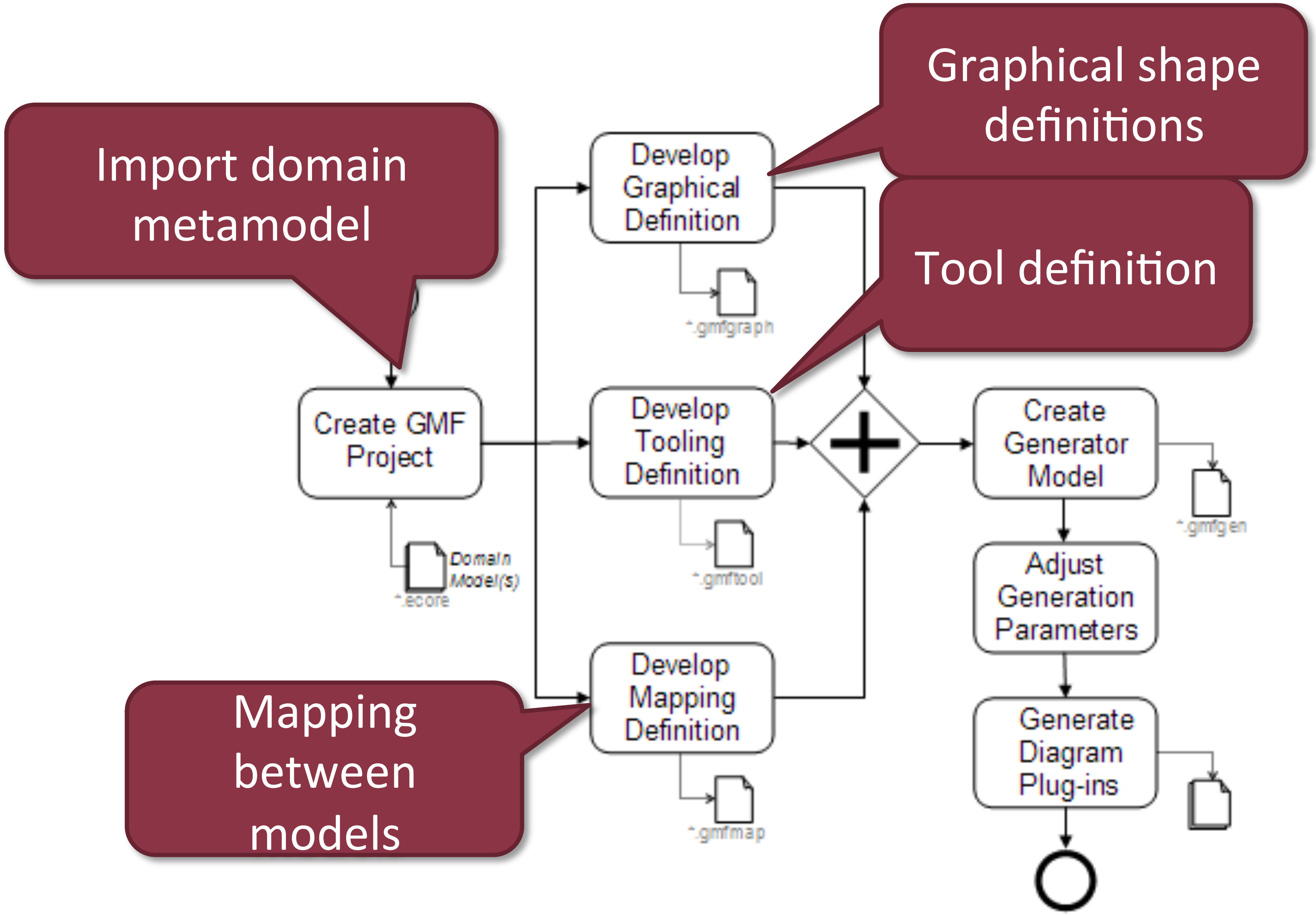
Mapping between models


Develop Graphical Definition
*.gmfgraph
Create GMF Project
Domain Model(s)
*.ecore
Develop Tooling Definition
*.gmftool
Develop Mapping Definition
*.gmfmap
Create Generator Model
*.gmfgen
Adjust Generation Parameters
Generate Diagram Plug-ins

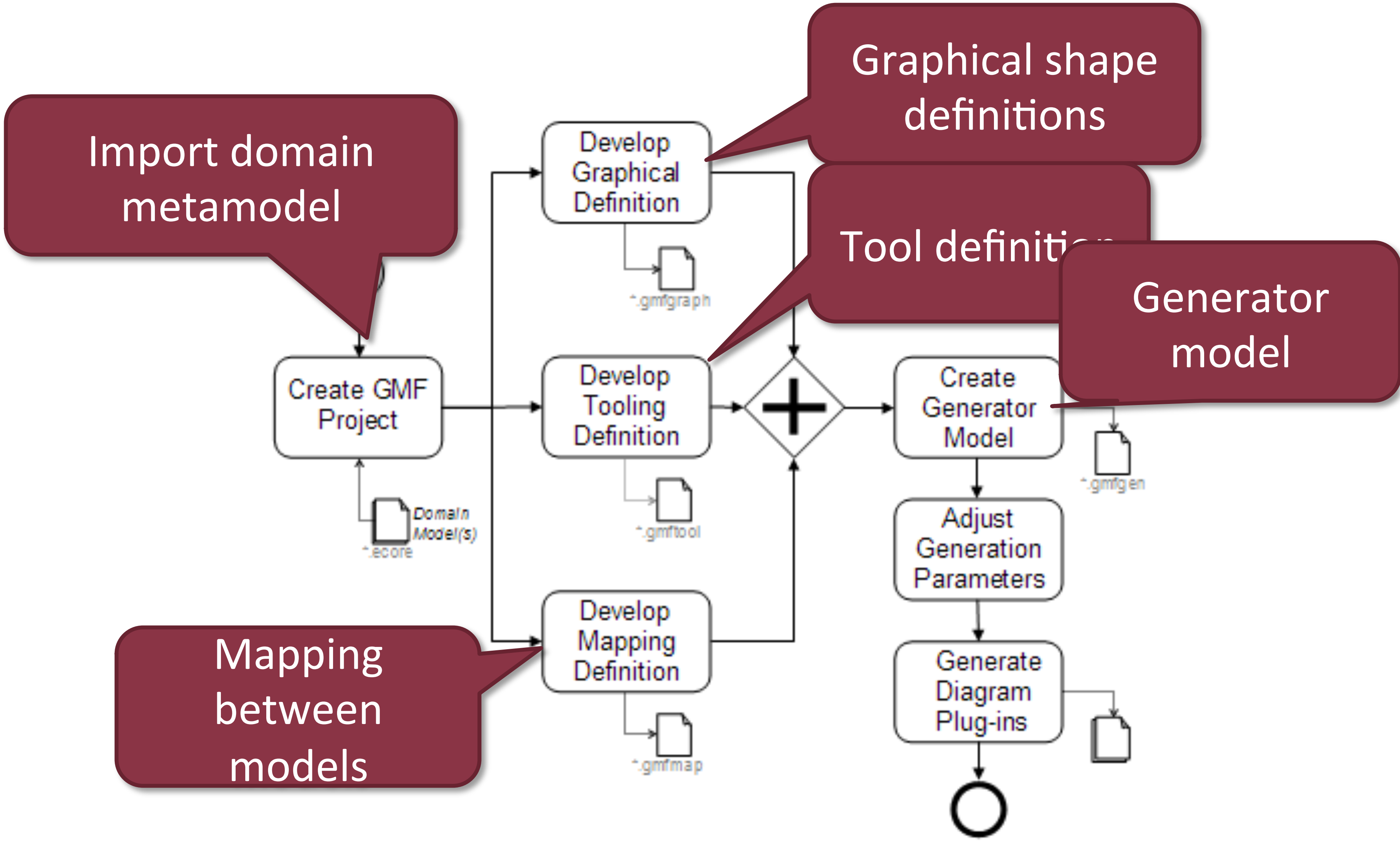
Generating GMF Editors


Graphical shape definitions
Import domain metamodel
Mapping between models
Develop Graphical Definition
*.gmfgraph
Create GMF Project
Develop Tooling Definition
*.gmftool
Domain Model(s)
*.ecore
Develop Mapping Definition
*.gmfmap
Create Generator Model
*.gmfgen
Adjust Generation Parameters
Generate Diagram Plug-ins

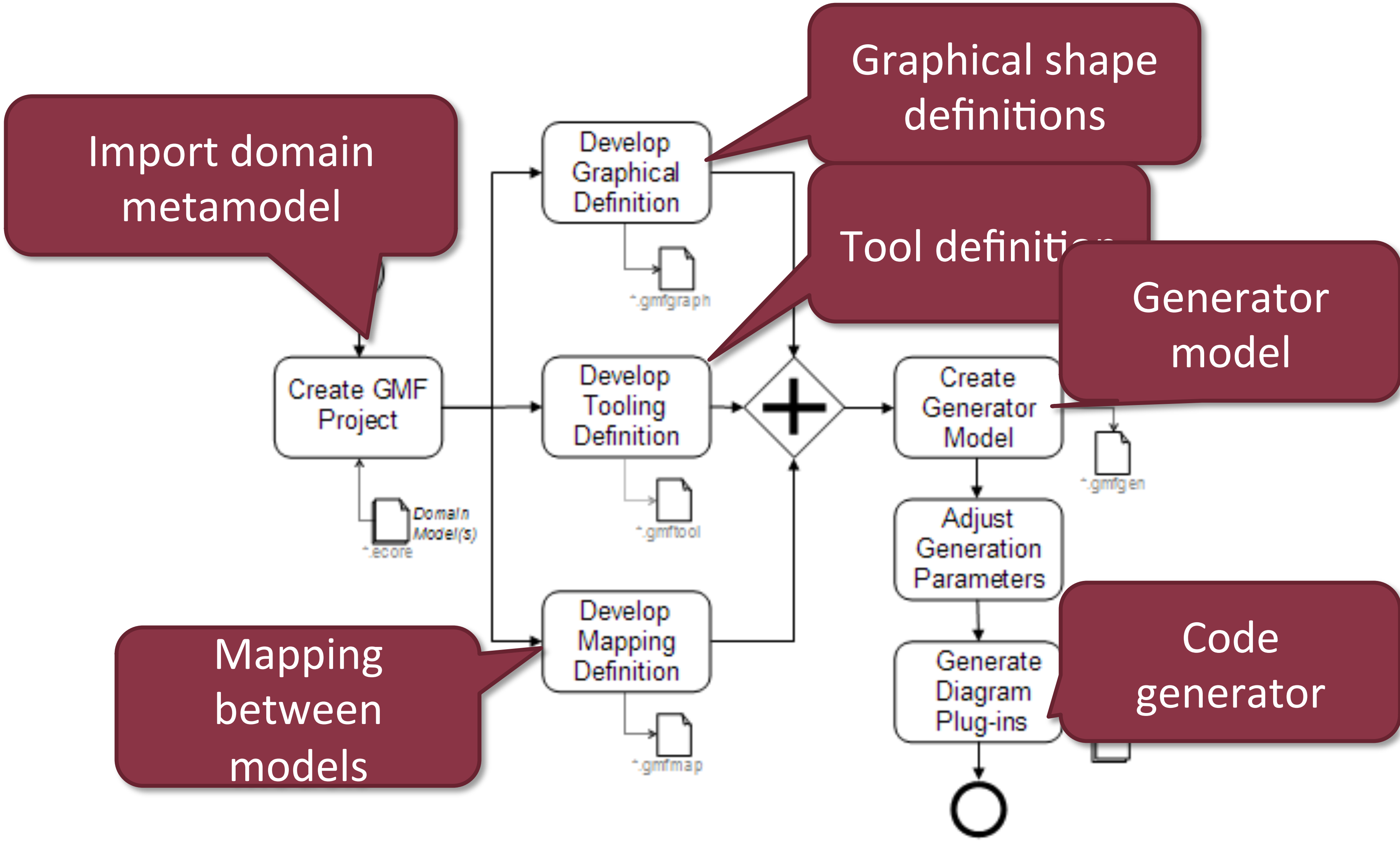
Generating GMF Editors


Import domain metamodel
Graphical shape definitions
Tool definition
Mapping between models
Develop Graphical Definition
*.gmfgraph
Create GMF Project
Domain Model(s)
*.ecore
Develop Tooling Definition
*.gmftool
Develop Mapping Definition
*.gmfmap
Create Generator Model
*.gmfgen
Adjust Generation Parameters
Generate Diagram Plug-ins

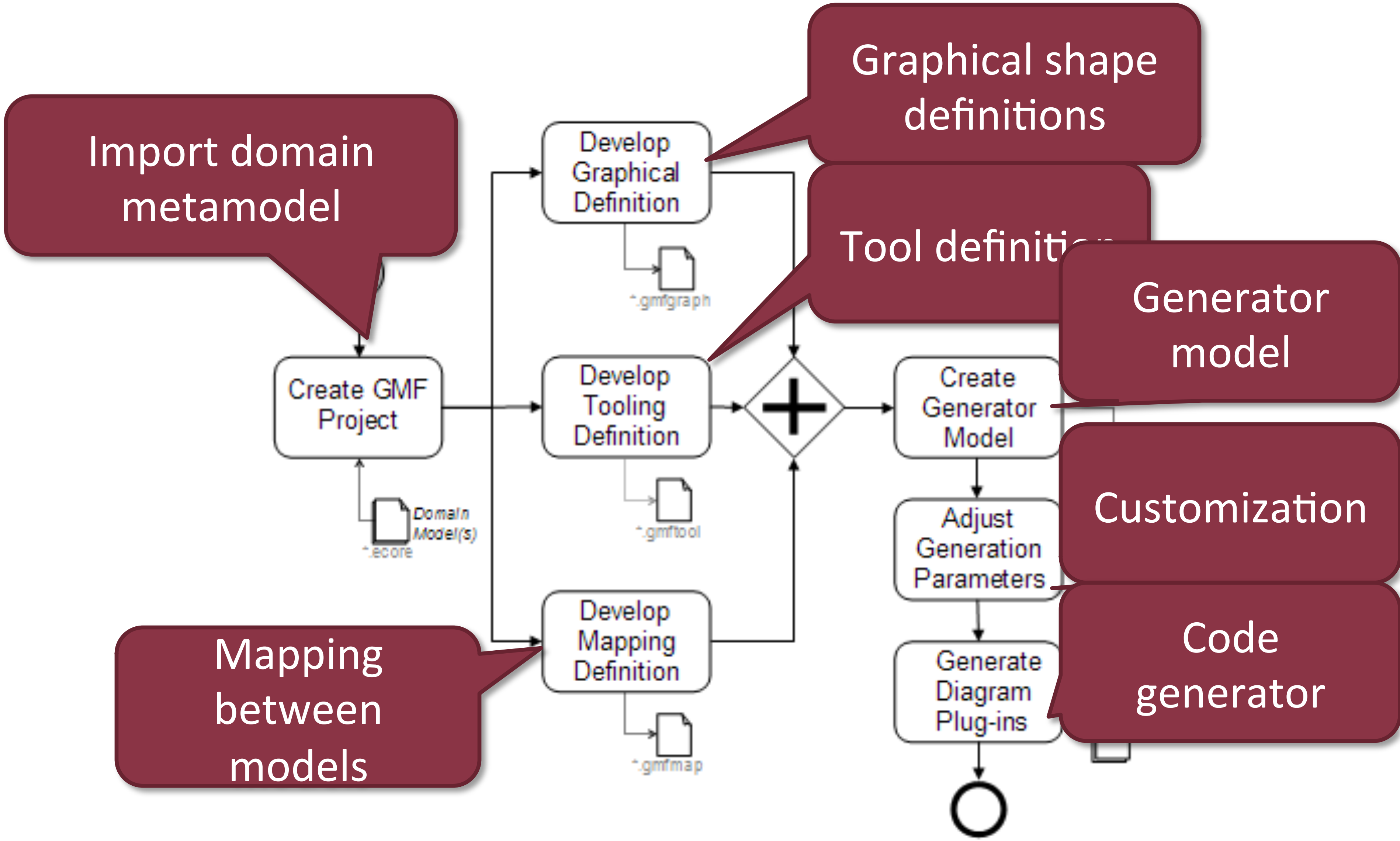
Generating GMF Editors



Generating GMF Editors

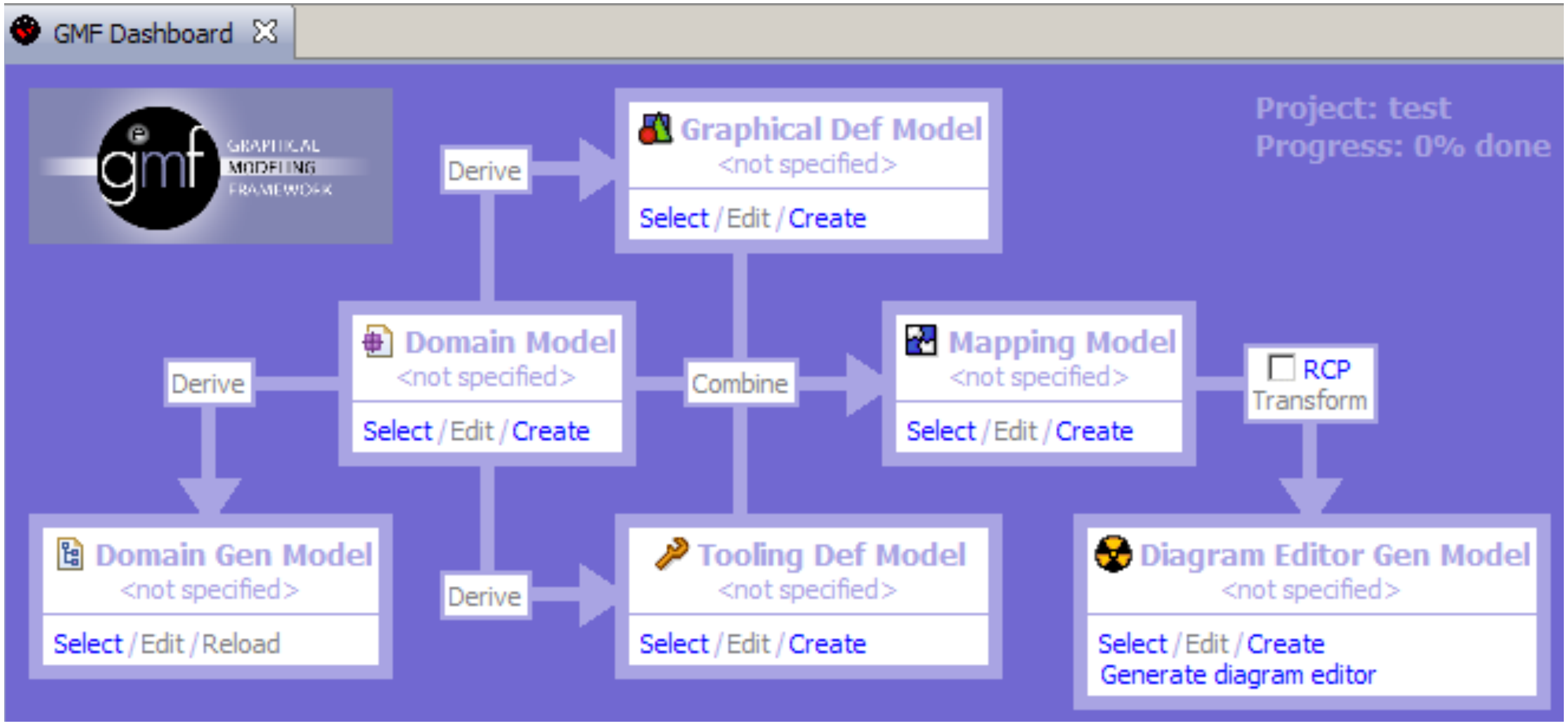


Generating GMF Editors



GMF Dashboard View

- To follow the editing process



GMF and EMF projects

- Required EMF projects for diagram editing
  - Model
  - Edit
- Recommended project structure
  - library – EMF project
    - model – Stored model files
      - library.ecore
      - library.genmodel

        EMF
      - library.gmfgraph
      - library.gmftool
      - library.gmfmap
      - library.gmfgen

        GMF
    - src – Generated EMF model code
  - library.diagram – Generated GMF project
  - library.edit – Generated EMF edit project

Domain Model

- Any kind of EMF model
    - Any EMF model editing technique works
- Good idea to have a single model root

Graphical Definition Model

- Goal
  - Specify used graphical model elements
  - Independent of domain model
- Define a figure library using
  - A tree editor
    - Not too simple
  - A wizard
    - Generates a model based on the domain model

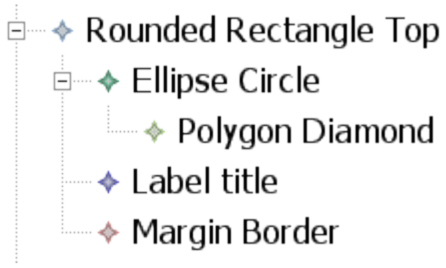
Graphical Definition Model

- Modeling instead of Java coding

Graphical Definition Model

- Modeling instead of Java coding

Create Figures on
predefined elements

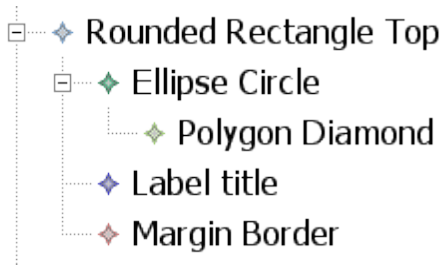


- Rounded Rectangle Top
  - Ellipse Circle
    - Polygon Diamond
  - Label title
  - Margin Border

Graphical Definition Model

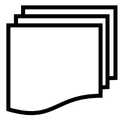
- Modeling instead of Java coding

Create Figures on
predefined elements



- Rounded Rectangle Top
  - Ellipse Circle
    - Polygon Diamond
  - Label title
  - Margin Border

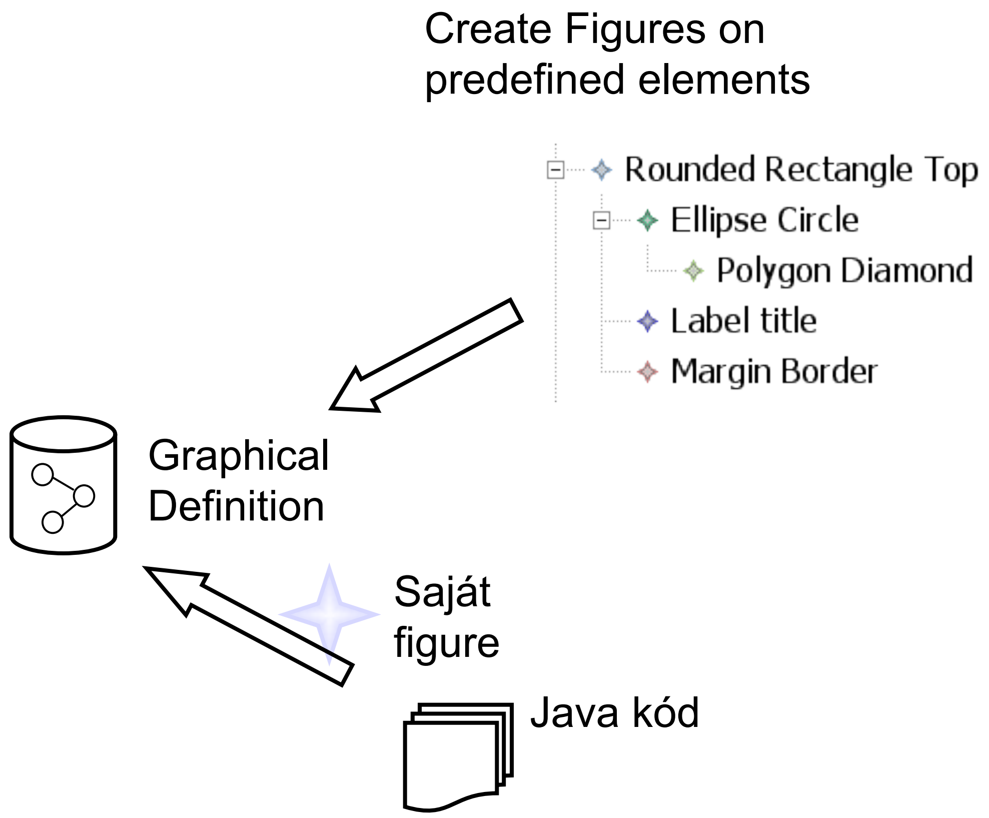
Saját
figure



Java kód

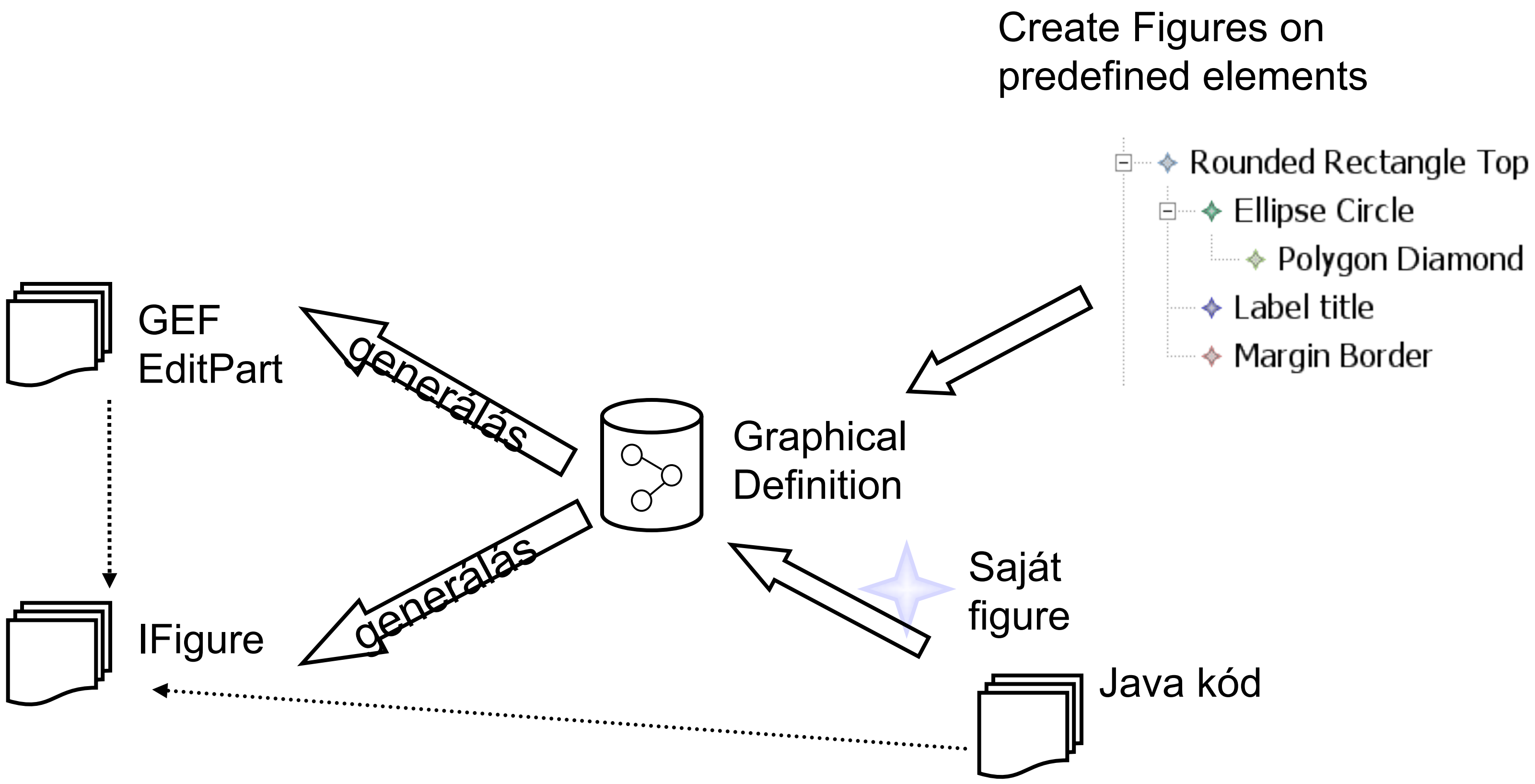
Graphical Definition Model

- Modeling instead of Java coding



Graphical Definition Model

- Modeling instead of Java coding



Graphical Definition Model

- Platform independent metamodel
- Elements:
  - Figure Galleries
    - Figure hierarchy
  - Nodes
  - Links
  - Compartments
  - Diagram Labels

Figure Gallery

- Figure descriptor
- Hierarchic figures
  - Label, Rectangle, Ellipse, Polygon, Polyline, Custom Figure stb.
  - Borders: Line, Margin, Compound, Custom
- Layouts
  - Flow, Border, Grid, XY, Stack, Custom
- Properties
  - Color, Font, Dimension, Insets
- Child Access: accessorok

Node

- Node type, diagram base element
- Refers to a Figure Descriptor
- Fill and border properties can be set

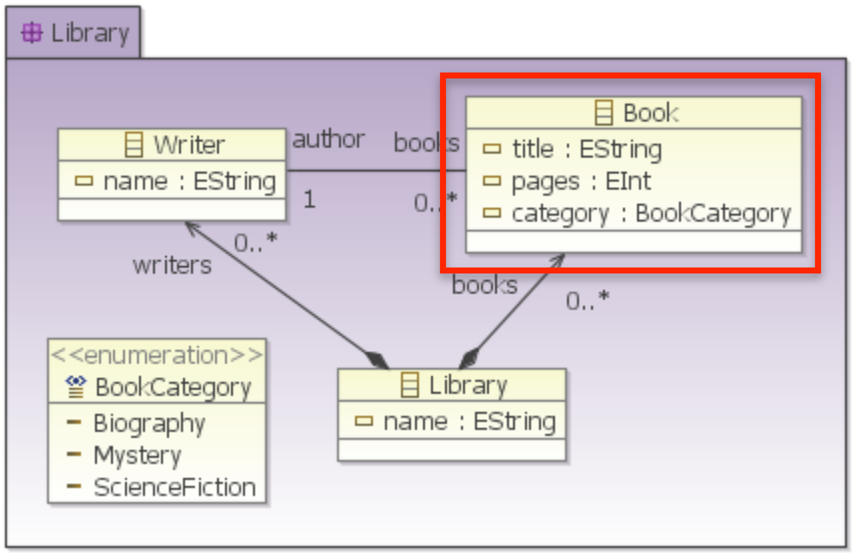
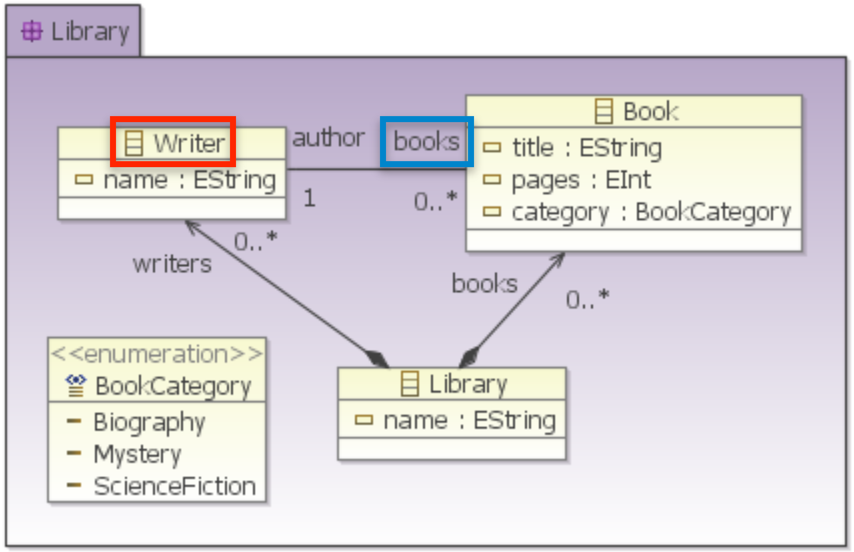


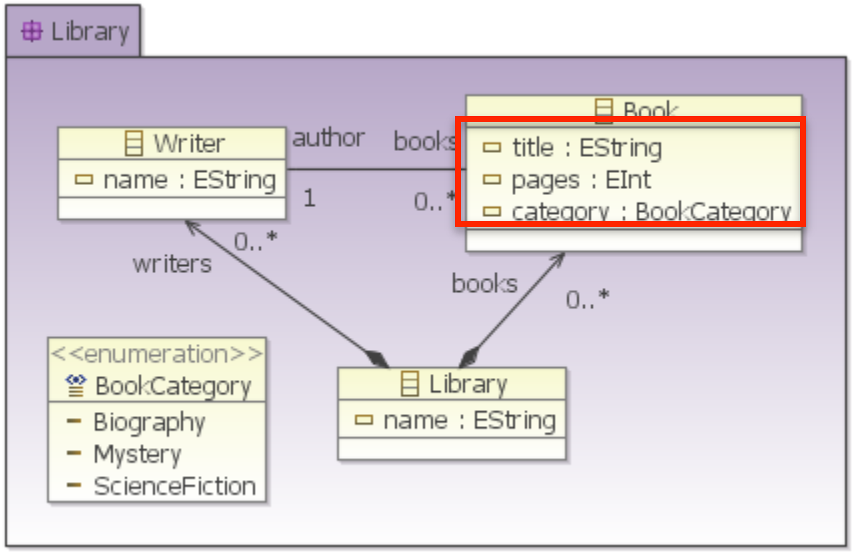
Diagram Label

- A label on the diagram
- Two types
  - Internal: Refers to a child access of a Figure descriptor
  - External: Refers to a Figure descriptor



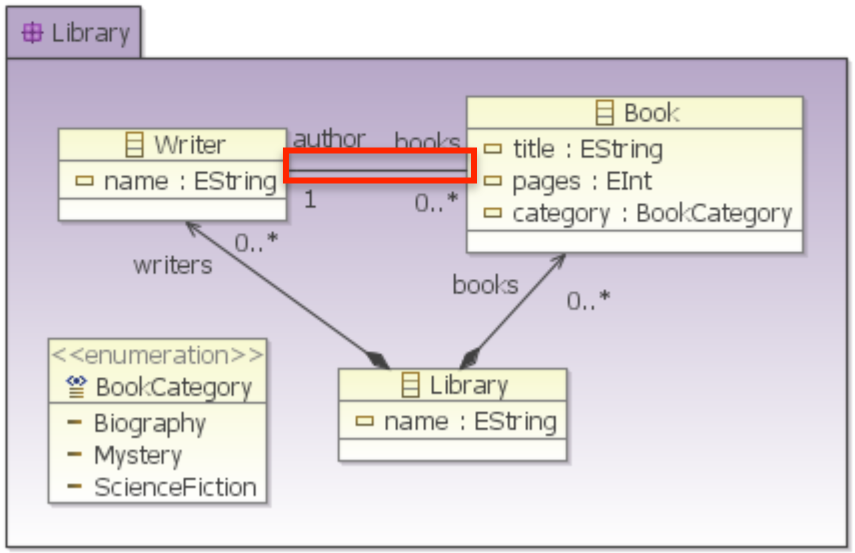
Compartment

- A „box” representing containment
- Refers to a Child access of the Figure descriptor
- Can be collapsed



Connection

- Edge on the graph
- Refers to a Figure descriptor



Tooling Definition Model

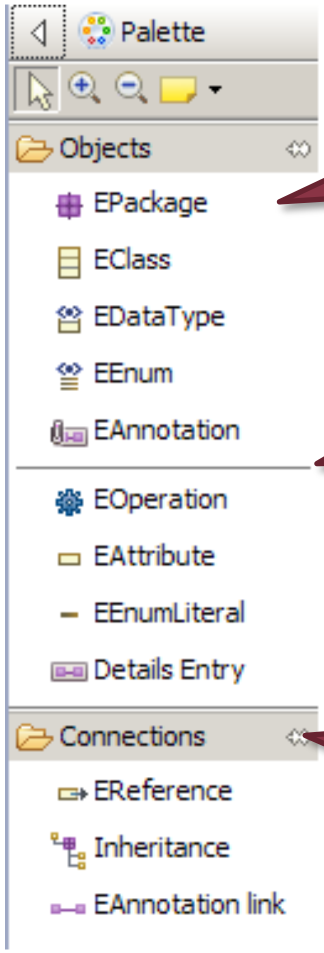
- Tooling metamodel (GMFTool)
- Definition of commands
- Wizard support
- Code generation generates commands

Tooling Definition Model

- Editing tools
  - Palette
  - Tool (typically creation)
    - Grouped into tool groups
  - Menu
    - Main/Popup
  - Action
  - Toolbar

- Base version can be generated

Palette



Tool

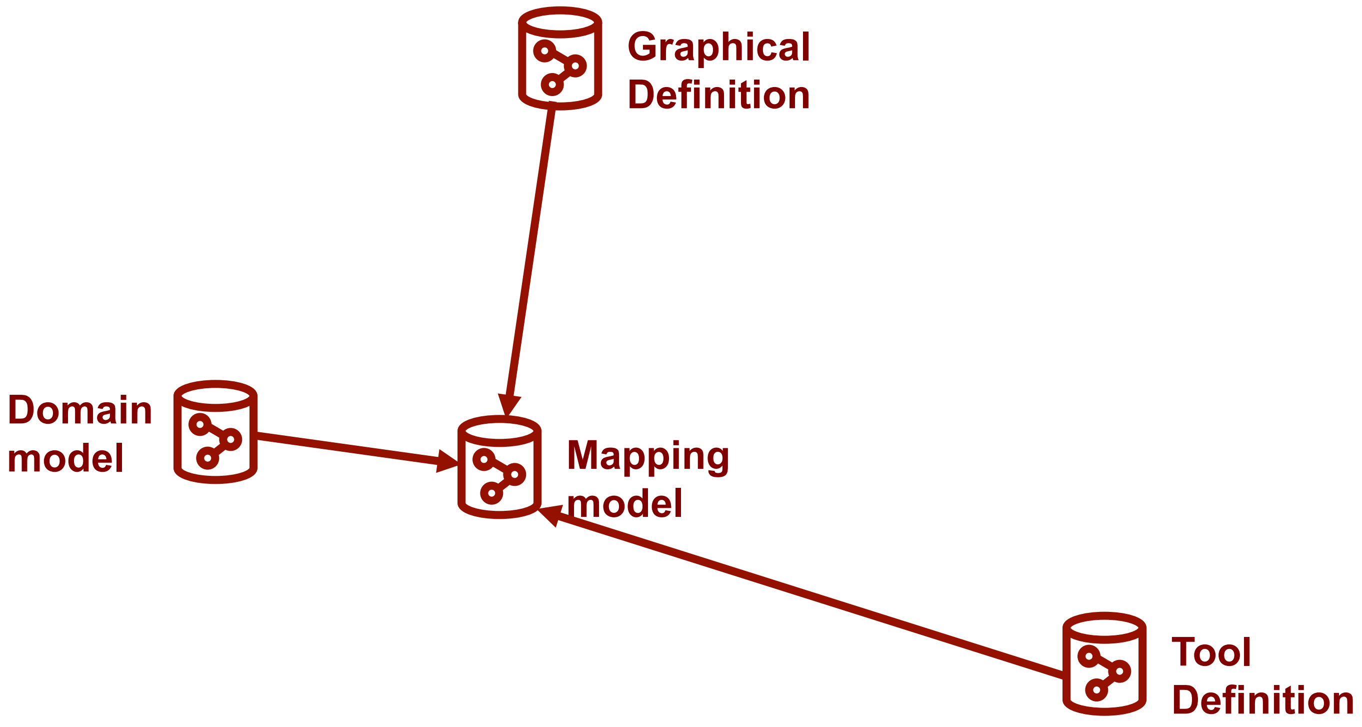
Palette Separator

Tool Group

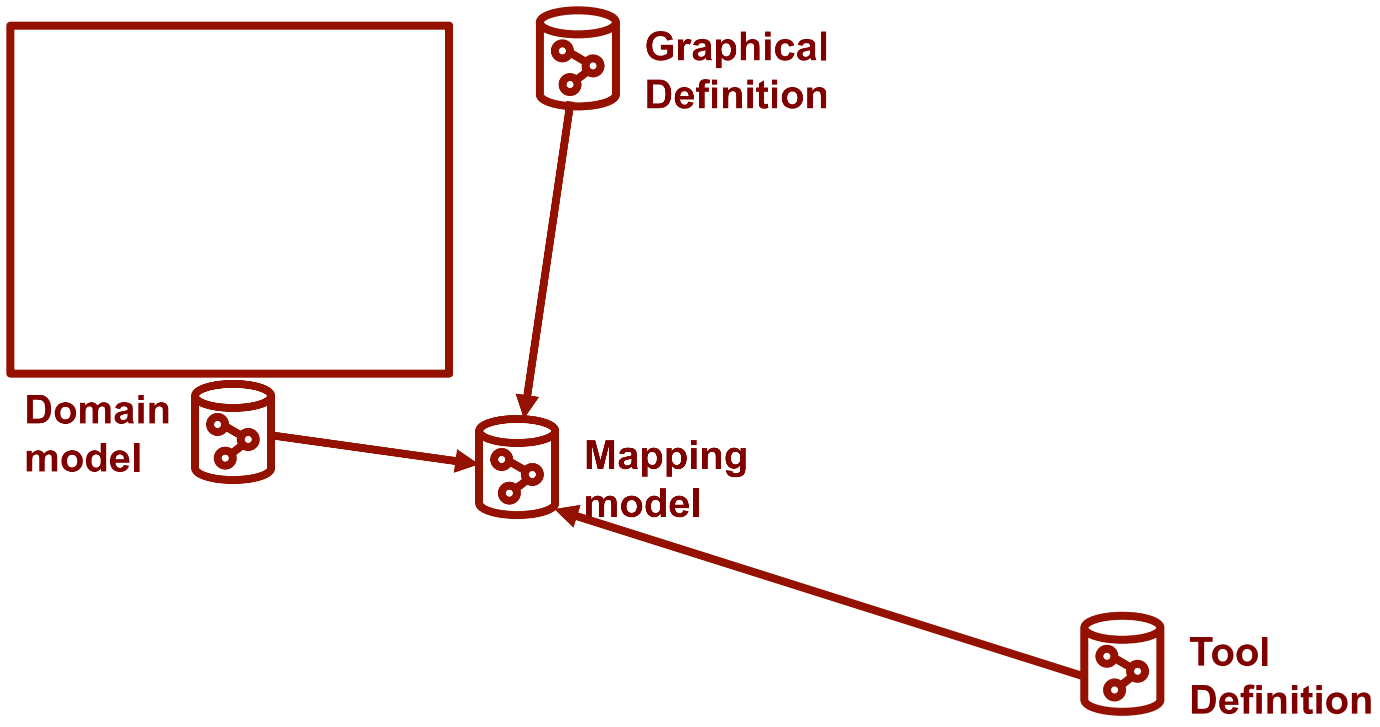
Mapping Model

- Mapping metamodel (GMFMap)
- Connects all previous models
- Defines mapping between the elements
- Domain – graphical – tooling

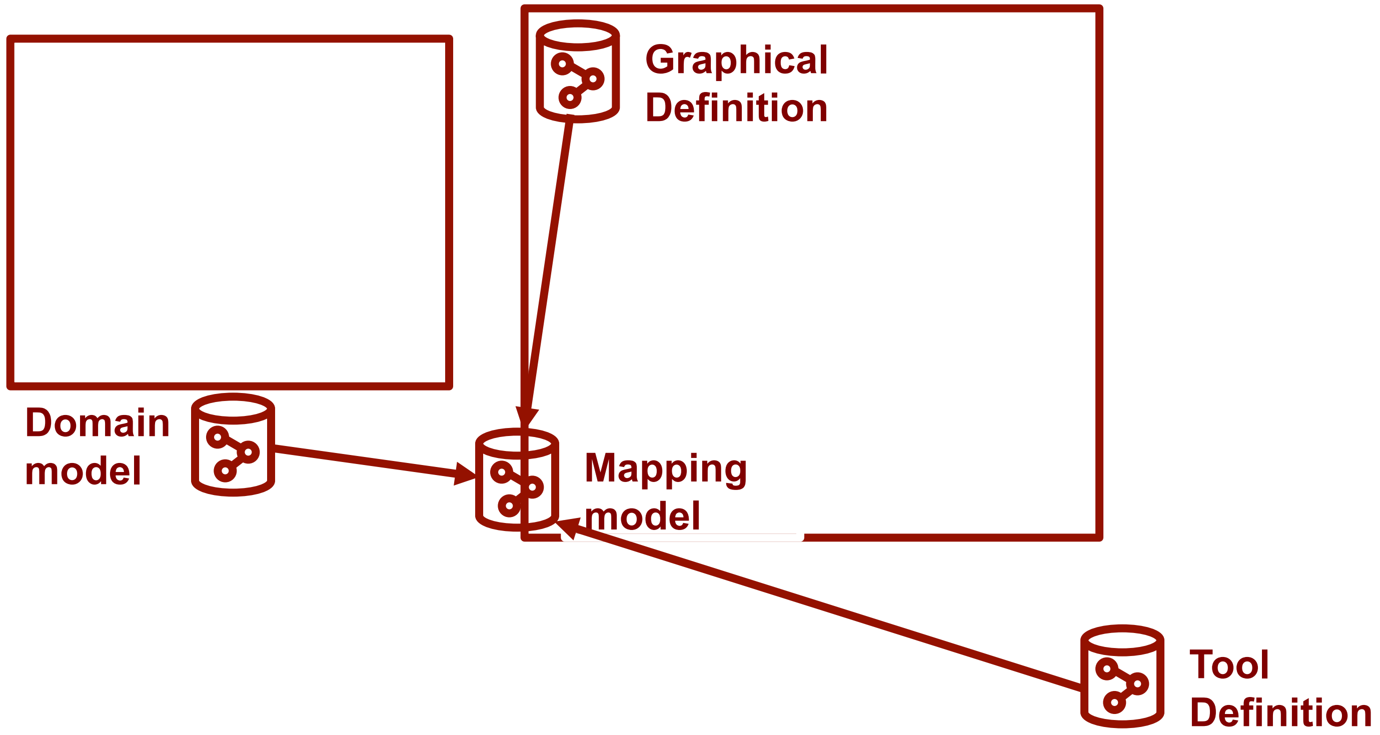
Mapping Model



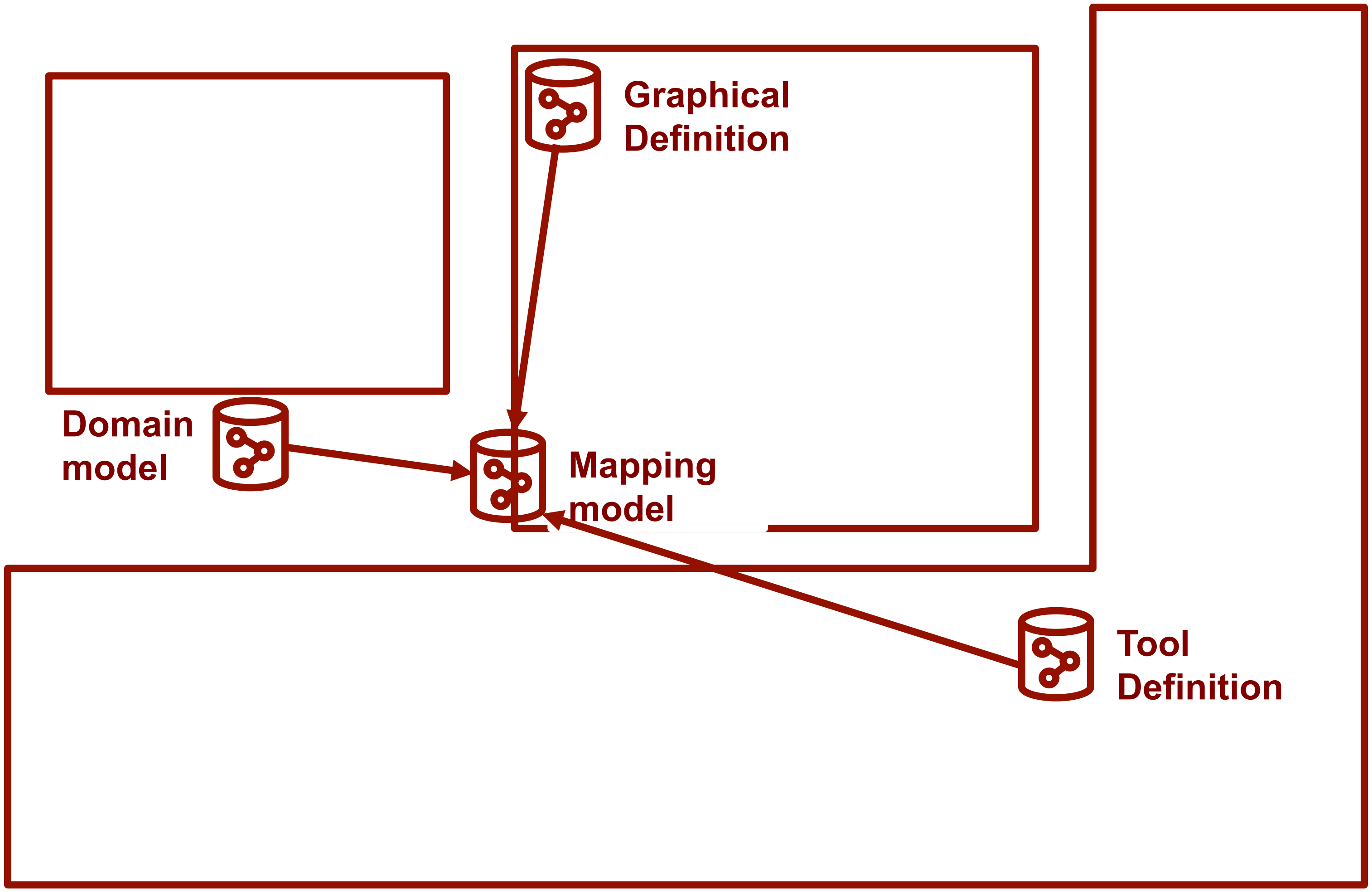
Mapping Model



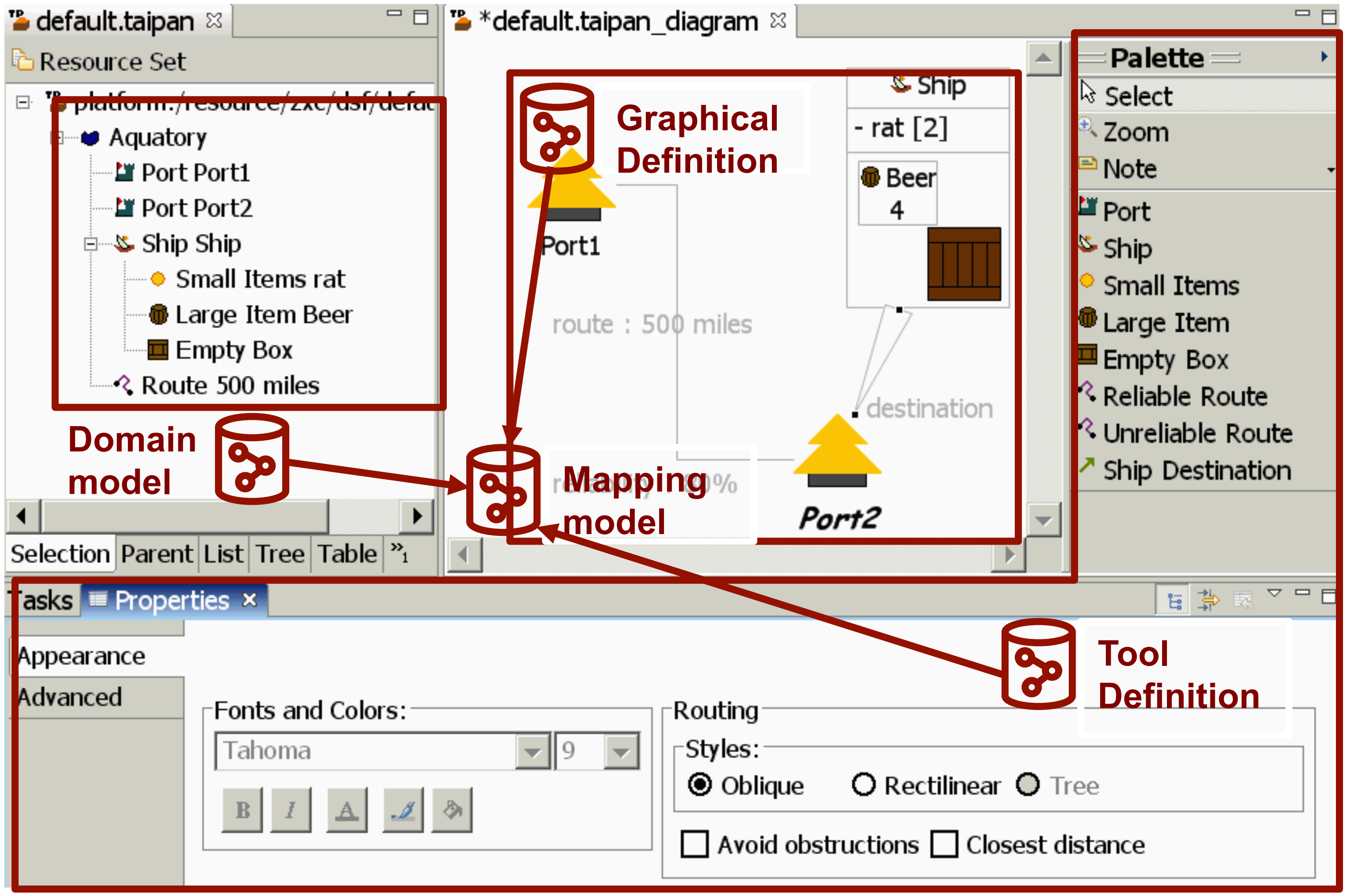
Mapping Model



Mapping Model



Mapping Model



Mapping Model

- Describes the logical structure between the
  - Domain model (.ecore)
  - Graphical model (.gmfgraph)
  - Tooling model (.gmftool)
- Can be validated
  - Model Validation
  - Constraint definition in OCL
- Base version generated based on domain model

Canvas Mapping

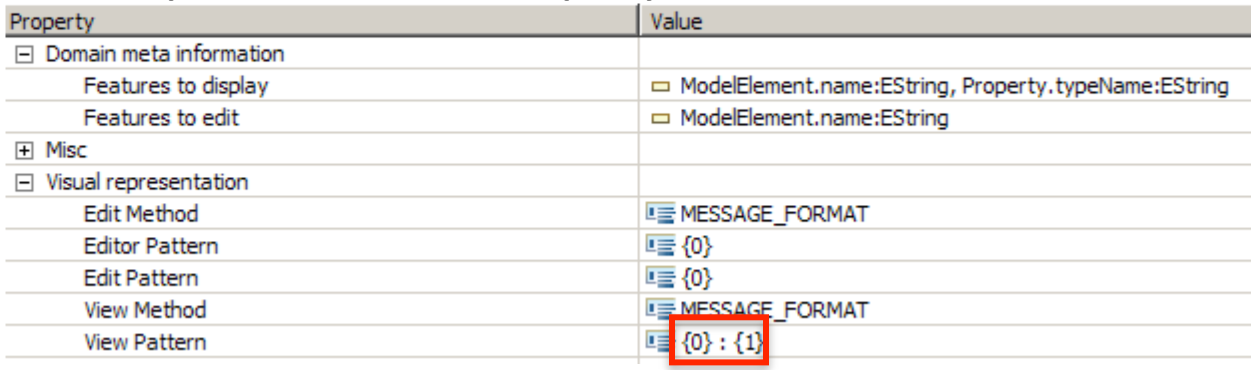
- Graphical: diagram „background” (gmfgraph Canvas root element)
- Domain: root of the model hierarchy
- Tooling:
  - Palette
  - Menus
  - Toolbar

Node Mapping

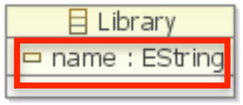
- Top Node Reference
  - Containment Feature: Selecting the containment feature of object represented by the Canvas Mapping
- Node Mapping
  - Graphical: diagram node
  - Domain: the class the node represents
  - Tooling: creation tool for the class
- Possible children
  - Label Mapping
  - Child Reference
  - Compartment Mapping

Label Mapping

- Graphical: Diagram label
- Domain:
  - (Design) Label Mapping: static text
  - Feature Label Mapping:
    - Features to display (and edit)
    - Textual patterns for display

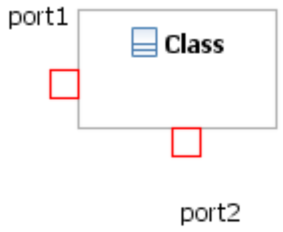


Property	Value
⊟ Domain meta information	
Features to display	ModelElement.name:EString, Property.typeName:EString
Features to edit	ModelElement.name:EString
⊞ Misc	
⊟ Visual representation	
Edit Method	MESSAGE_FORMAT
Editor Pattern	{0}
Edit Pattern	{0}
View Method	MESSAGE_FORMAT
View Pattern	{0} : {1}



Child Reference

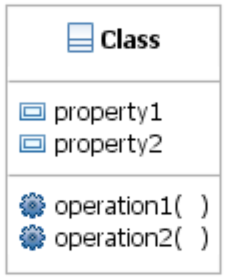
- Children of a node
- Required data:
  - Containment Feature and Node Mapping
- Two type:
  - Affixed: displayed outside the node (e.g. ports)



  - Compartment: children displayed in a compartment

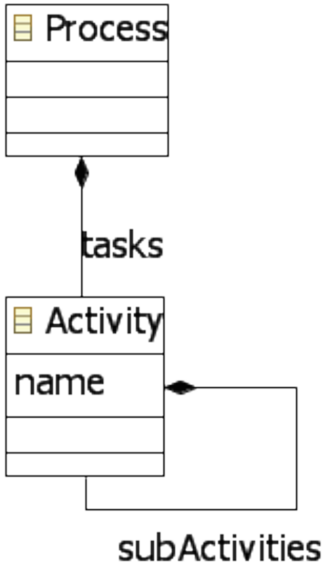
Compartment Mapping

- Graphical: Compartment formal description
  - Child Reference
  - A compartment can only contain elements of the same type

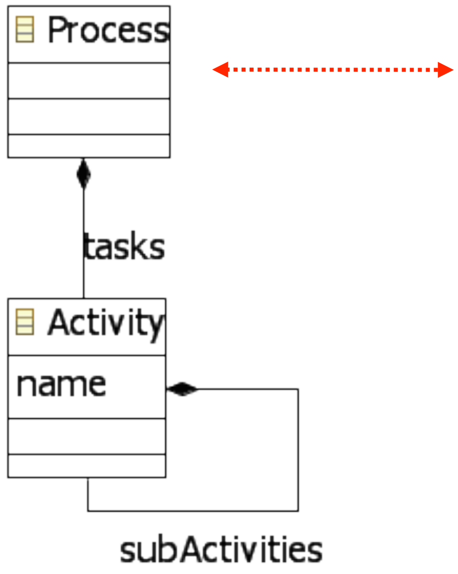


Node hierarchy – example

Node hierarchy – example

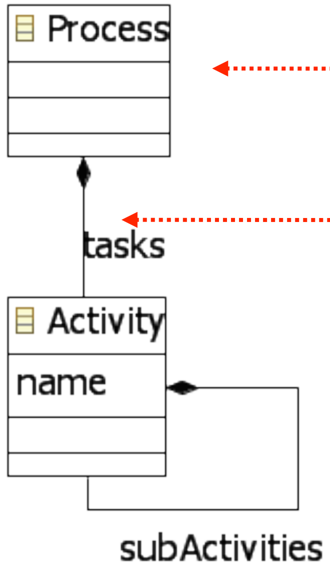


Node hierarchy – example



Canvas Mapping

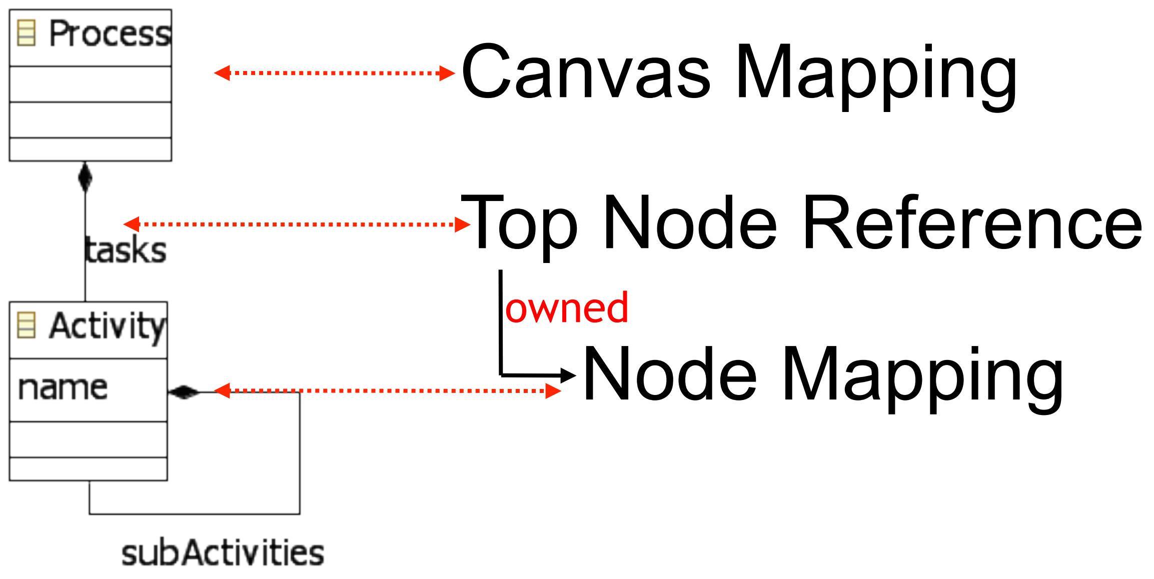
Node hierarchy – example



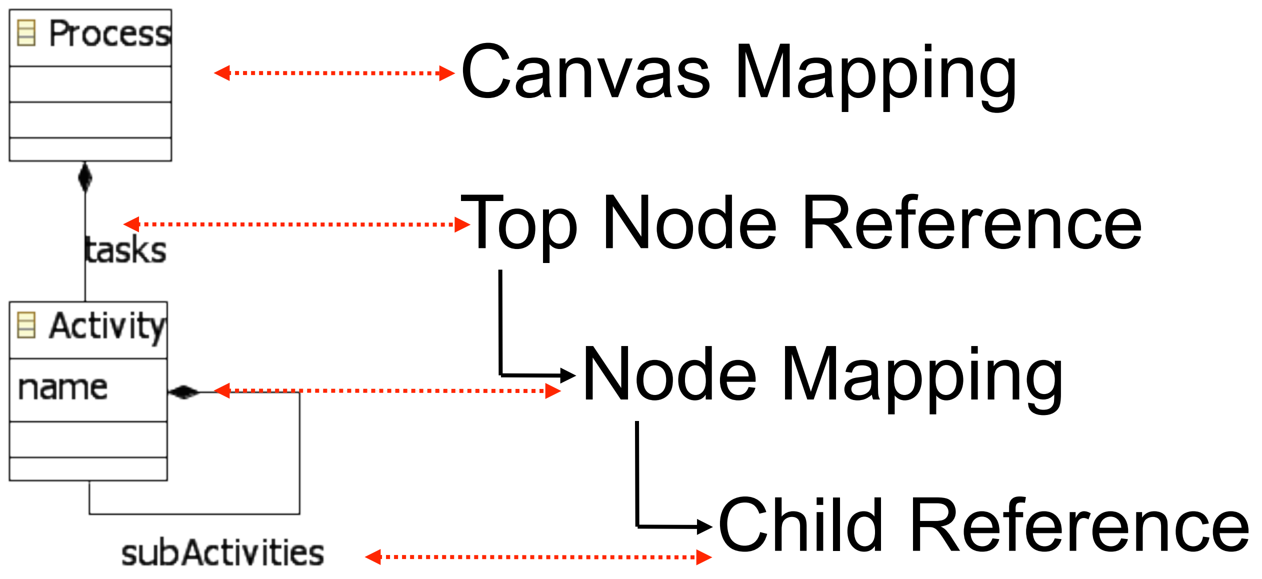
Canvas Mapping

Top Node Reference

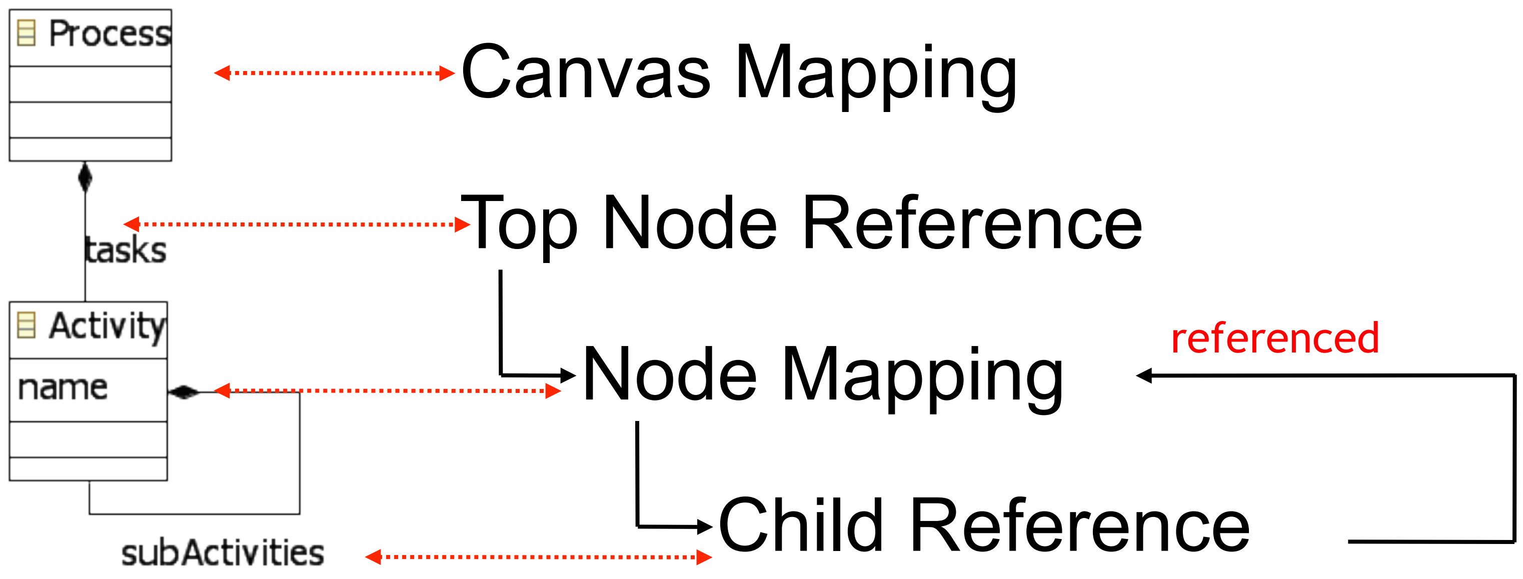
Node hierarchy – example



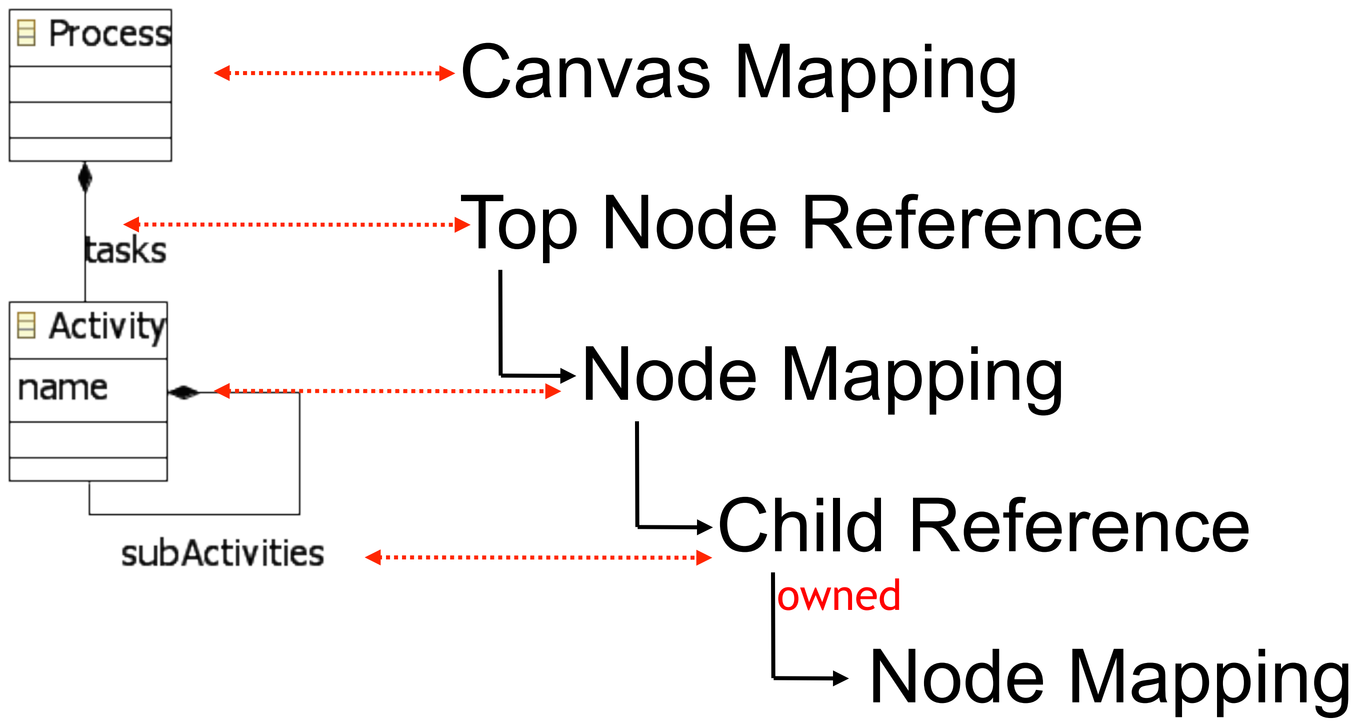
Node hierarchy – example



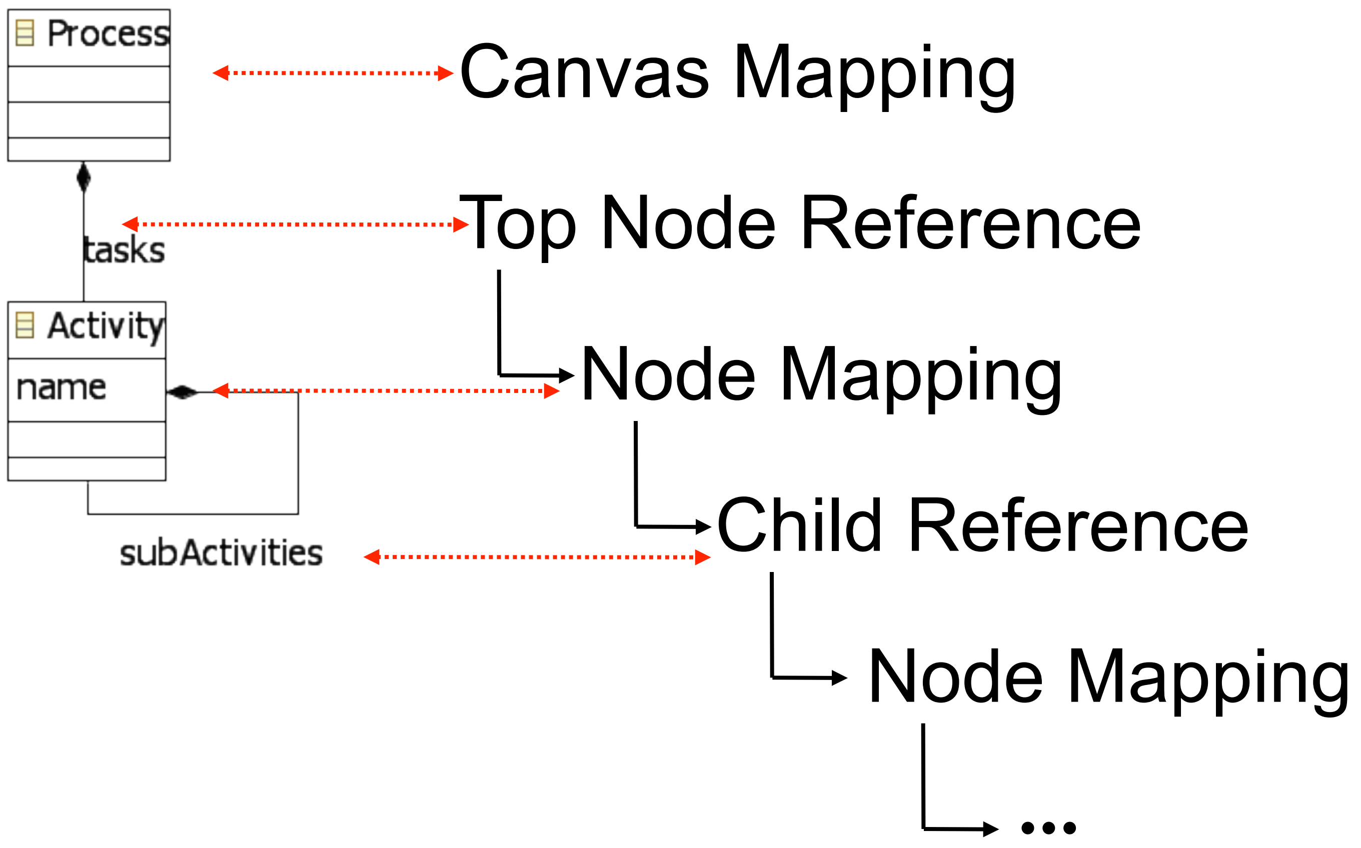
Node hierarchy – example



Node hierarchy – example



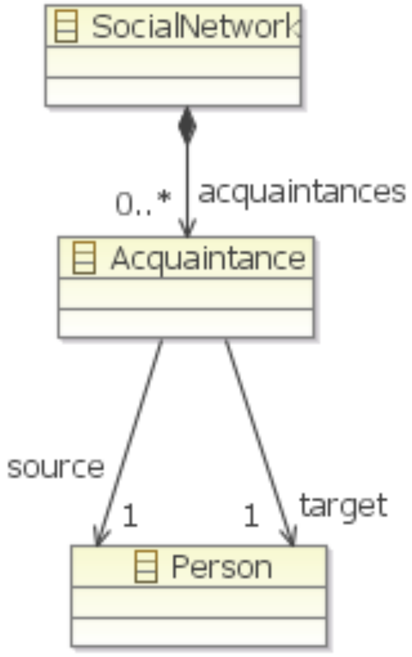
Node hierarchy – example

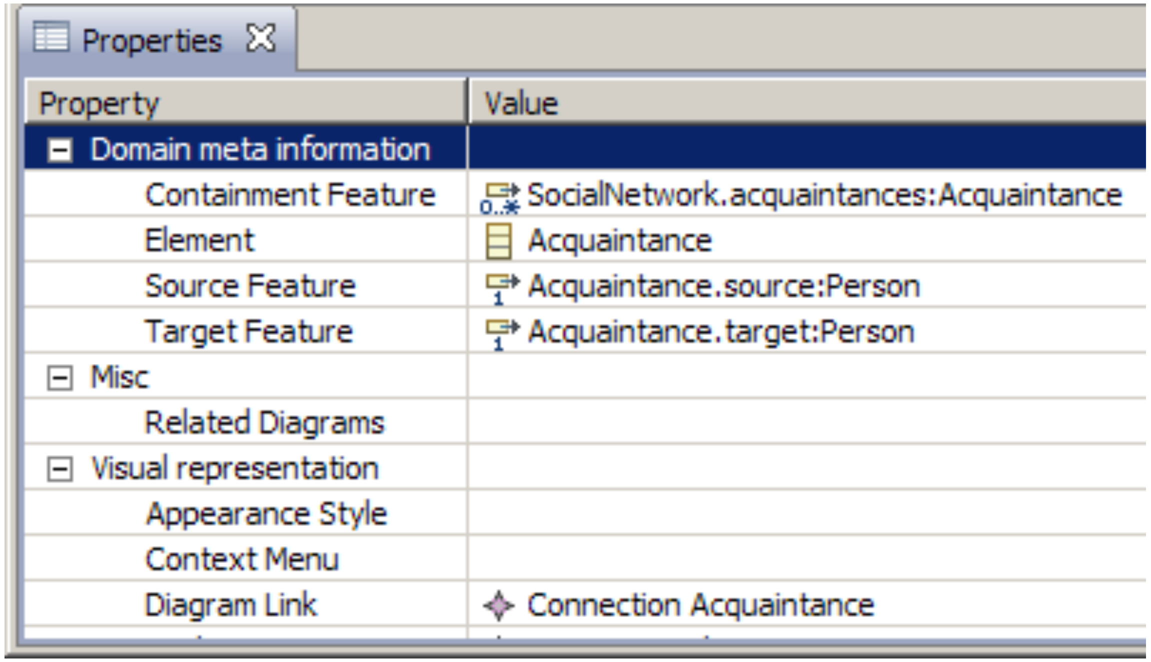


Link Mapping

- Graphical: connection
- Domain:
  - Connection is a feature:
    - Set up in Target Feature
  - Connection is a class
    - Element: the representation class
    - Containment Feature
    - Source/Target Feature: two ends of a connection
- Tooling: creation tool for the connection

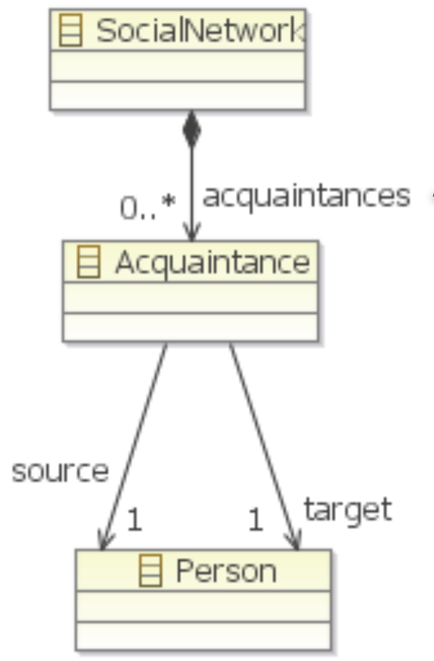
Connection via class - example

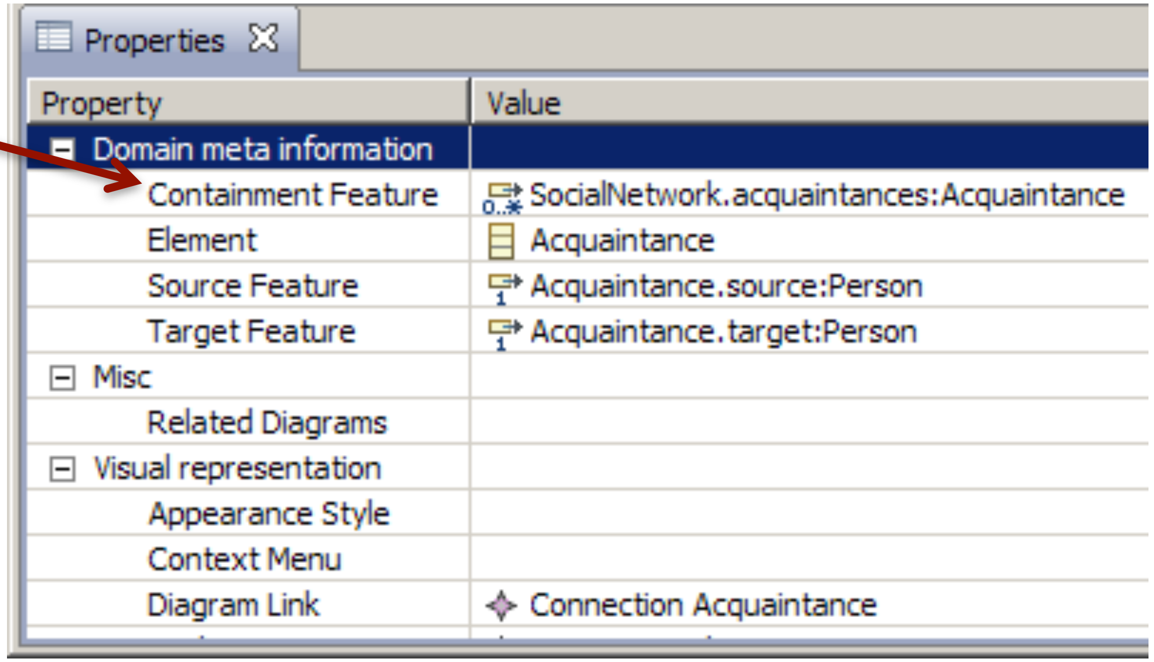




Property	Value
⊟ Domain meta information	
Containment Feature	SocialNetwork.acquaintances:Acquaintance
Element	Acquaintance
Source Feature	Acquaintance.source:Person
Target Feature	Acquaintance.target:Person
⊟ Misc	
Related Diagrams	
⊟ Visual representation	
Appearance Style	
Context Menu	
Diagram Link	Connection Acquaintance

Connection via class - example

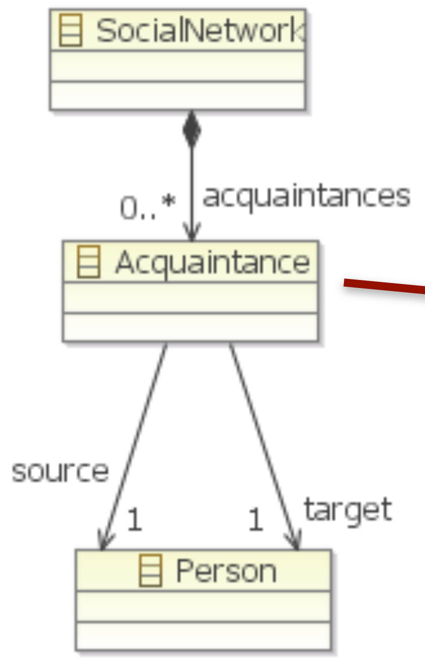


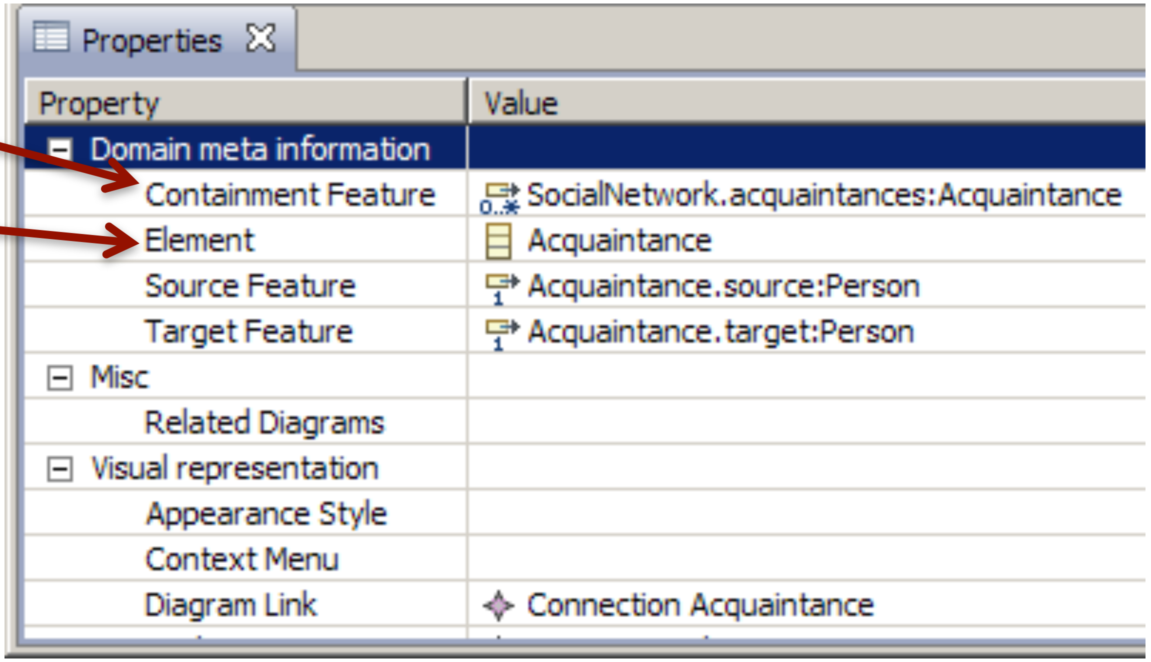


Properties

Property	Value
Domain meta information	
Containment Feature	SocialNetwork.acquaintances:Acquaintance
Element	Acquaintance
Source Feature	Acquaintance.source:Person
Target Feature	Acquaintance.target:Person
Misc	
Related Diagrams	
Visual representation	
Appearance Style	
Context Menu	
Diagram Link	Connection Acquaintance

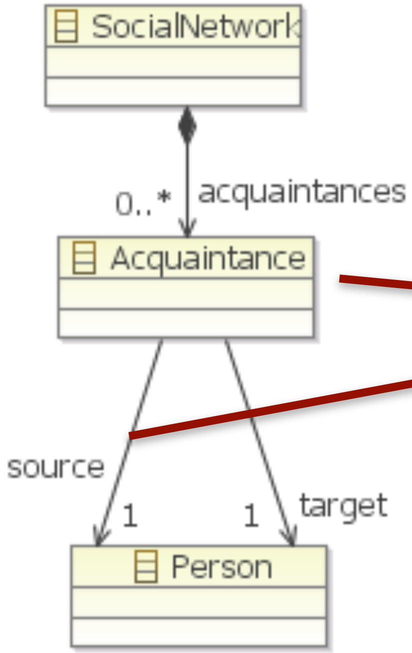
Connection via class - example

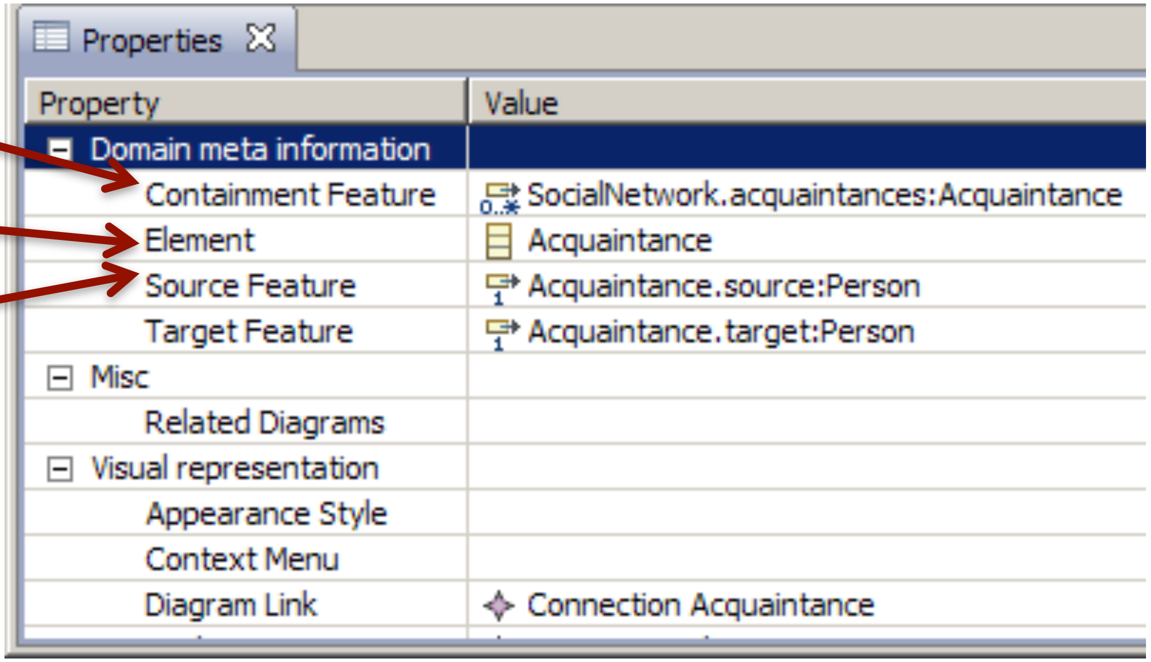




Property	Value
Domain meta information	
Containment Feature	SocialNetwork.acquaintances:Acquaintance
Element	Acquaintance
Source Feature	Acquaintance.source:Person
Target Feature	Acquaintance.target:Person
Misc	
Related Diagrams	
Visual representation	
Appearance Style	
Context Menu	
Diagram Link	Connection Acquaintance

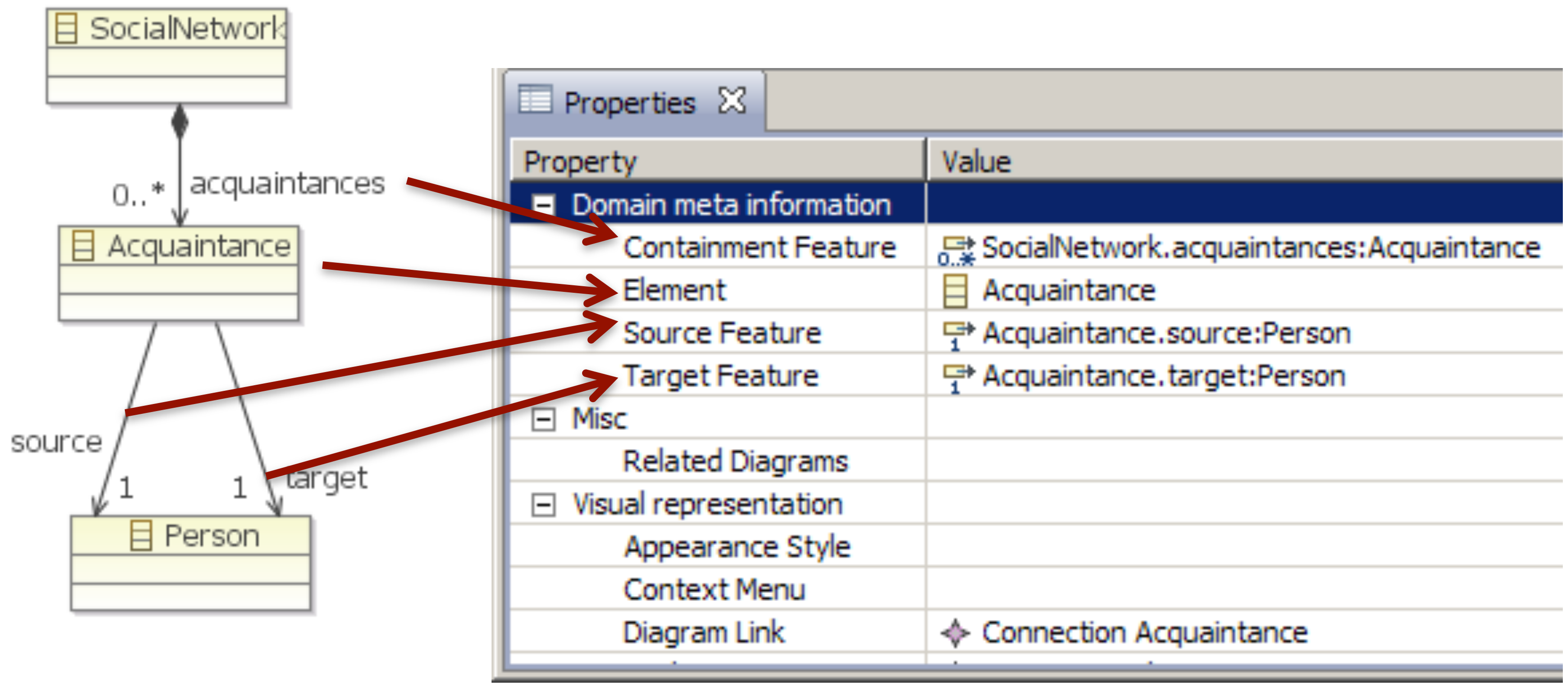
Connection via class - example




Properties

Property	Value
Domain meta information	
Containment Feature	SocialNetwork.acquaintances:Acquaintance
Element	Acquaintance
Source Feature	Acquaintance.source:Person
Target Feature	Acquaintance.target:Person
Misc	
Related Diagrams	
Visual representation	
Appearance Style	
Context Menu	
Diagram Link	Connection Acquaintance

Connection via class - example



Property	Value
⊟ Domain meta information	
Containment Feature	SocialNetwork.acquaintances:Acquaintance
Element	Acquaintance
Source Feature	Acquaintance.source:Person
Target Feature	Acquaintance.target:Person
⊟ Misc	
Related Diagrams	
⊟ Visual representation	
Appearance Style	
Context Menu	
Diagram Link	Connection Acquaintance

Diagram Editor Generator Model

- Generator model (GMFGen)
- Code generation parameters
  - Similar to EMF genmodel
  - Created by transforming the mapping model
- Code generation via Java Emitter Templates/Xpand/ Xtend2
  - Replaceable templates
  - Target platform: GMF Runtime
- Configurable generation
  - Plug-in ID, provider name, package namespace, etc.
- Runtime options
  - Print support, validation, etc.
  - Diagram persistence

Generator information

- File properties (Gen Editor)
  - Model and diagram file extension
  - Separation of model and diagram file
- Plugin identifier information (Gen Plugin)
  - ID, name, provider
- Additional edit capabilities – (Gen Diagram)
  - Validation
  - Shortcuts
  - Providers

Generated code

- Complete editor code
  - Based on GEF and EMF
  - With GMF Runtime features

Process similar to EMF...

Process similar to EMF…

EMF

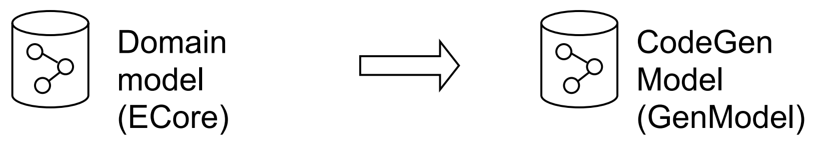
Process similar to EMF...

EMF

Domain model (ECore)

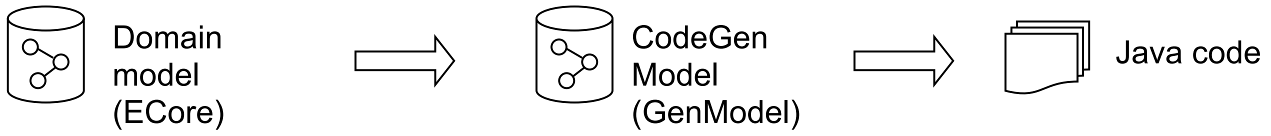
Process similar to EMF…

EMF



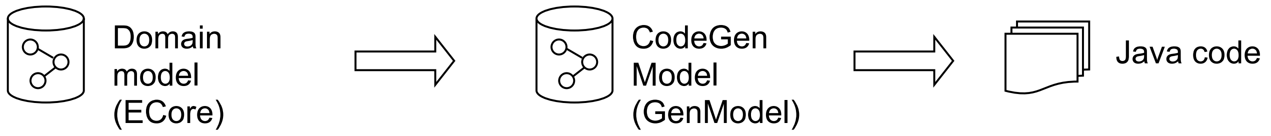
Process similar to EMF...

EMF



Process similar to EMF...

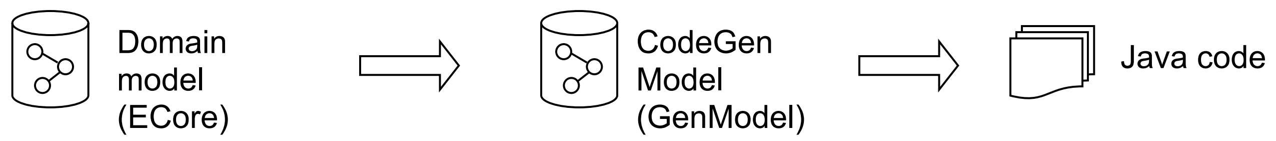
EMF



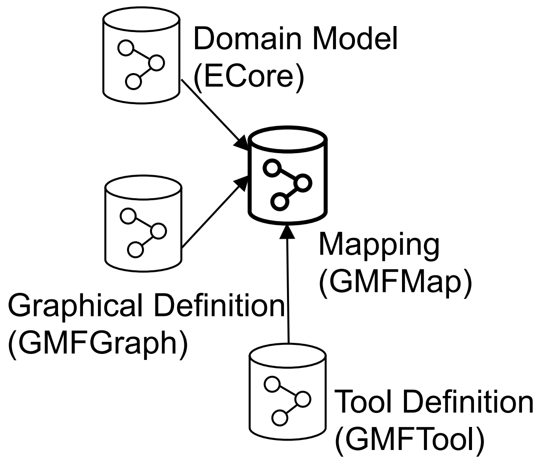
GMF

Process similar to EMF...

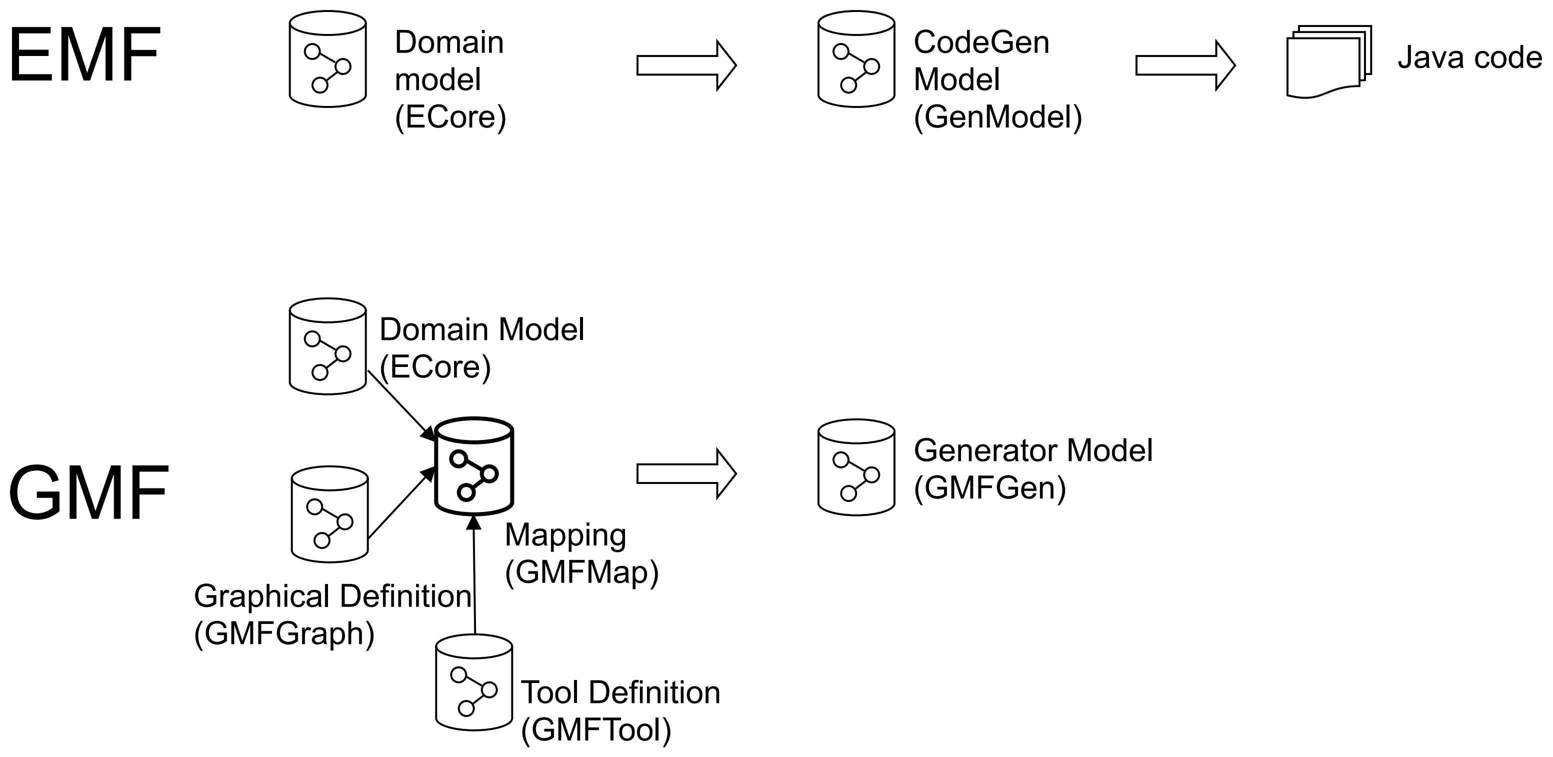
EMF



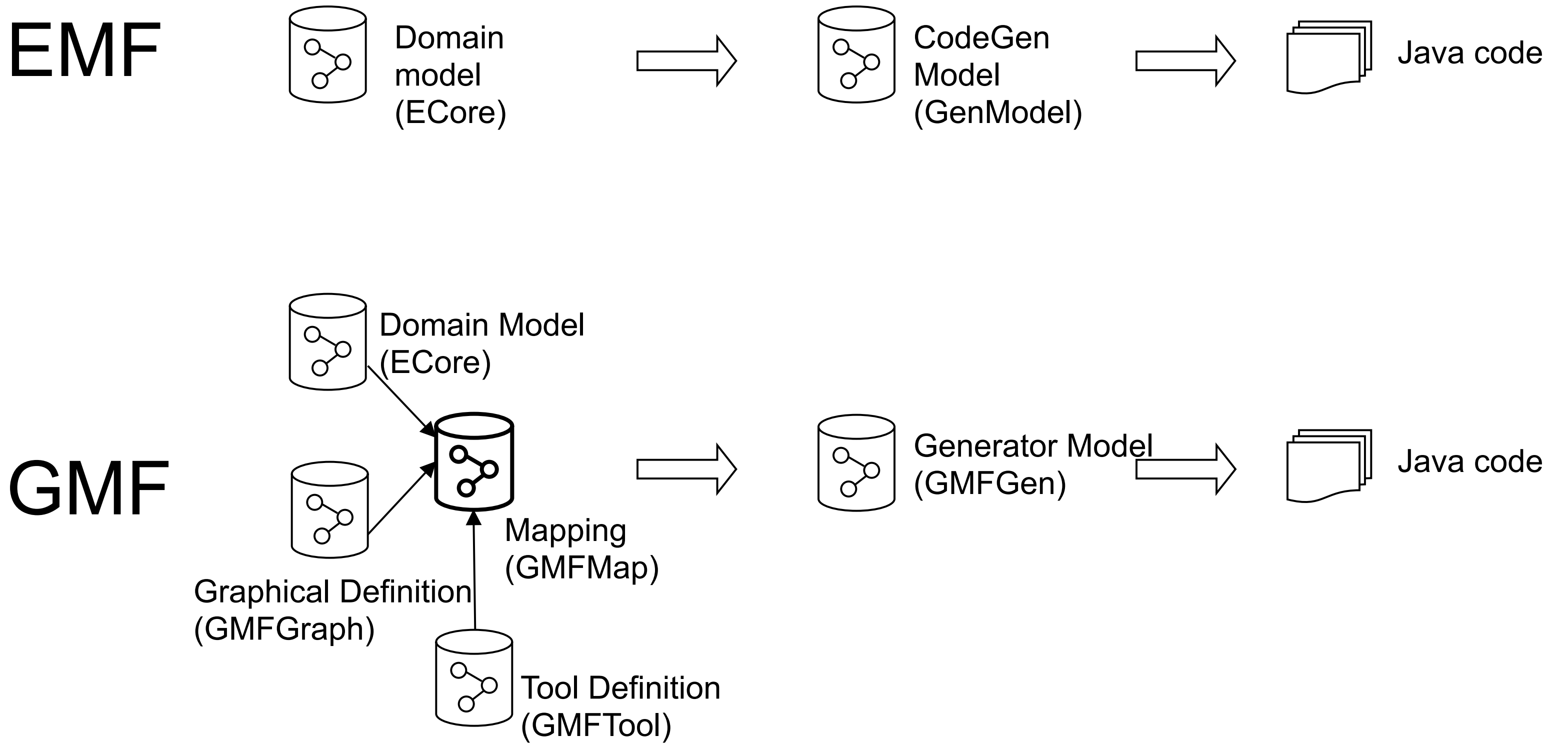
GMF



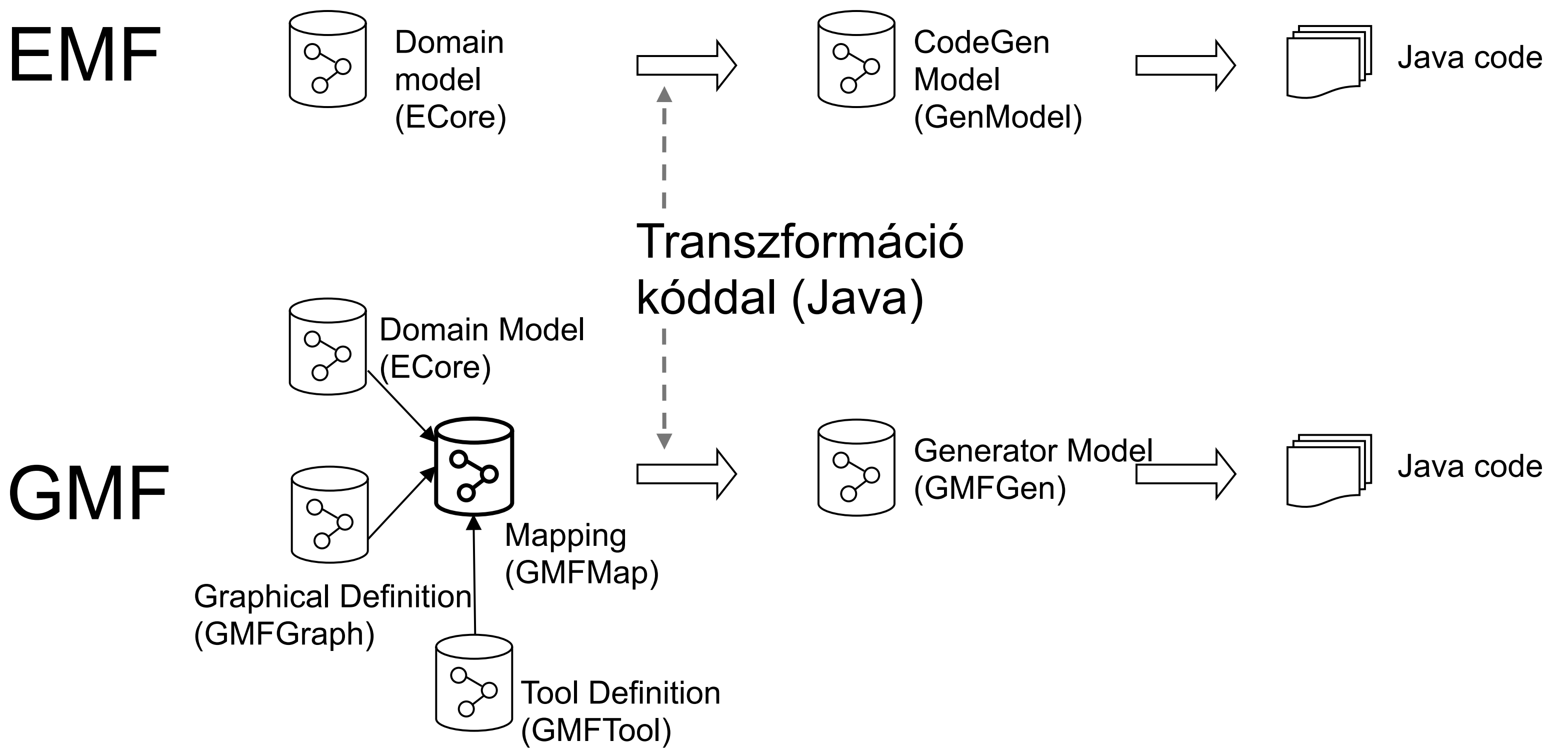
Process similar to EMF…



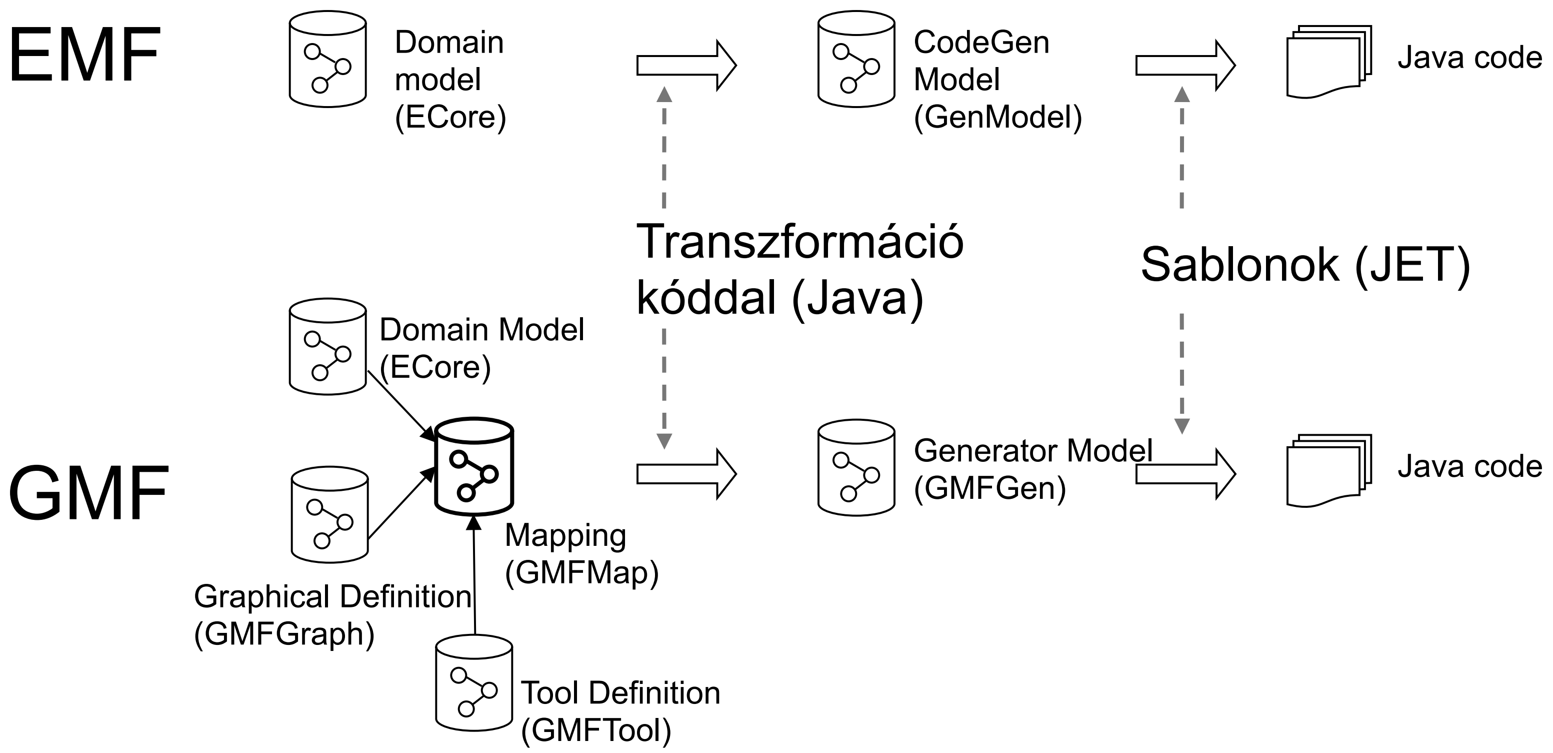
Process similar to EMF...



Process similar to EMF...



Process similar to EMF...



Customizing generated code

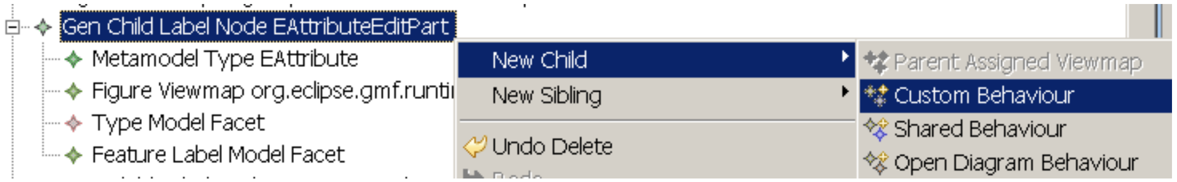
- Possibilities:
  - Custom classes in tooling model
  - Code overwrite
  - Template extension
  - Extension points

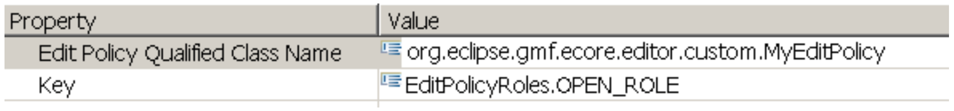
Customization – Manual classes

- Graphical
  - Figure, Connection, Decoration, Border, Layout
- Generator
  - Behaviour (e.g. double click handling)
- Advantages
  - Model driven approach (incl. attributes)
- Disadvantages
  - Refactoring, statikus validation problematic because of abstraction differences

Customization – Manual classes

- Example

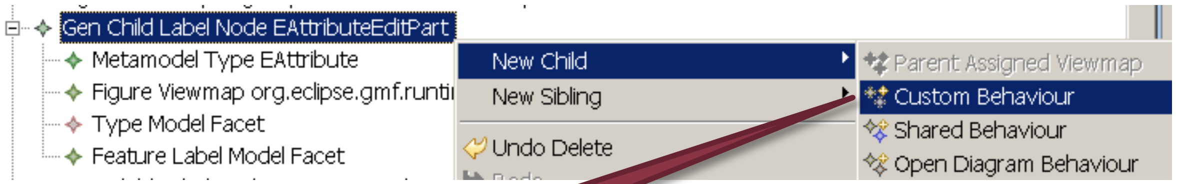

Gen Child Label Node EAttributeEditPart
Metamodel Type EAttribute
Figure Viewmap org.eclipse.gmf.runti
Type Model Facet
Feature Label Model Facet
New Child
New Sibling
Undo Delete
Parent Assigned Viewmap
Custom Behaviour
Shared Behaviour
Open Diagram Behaviour



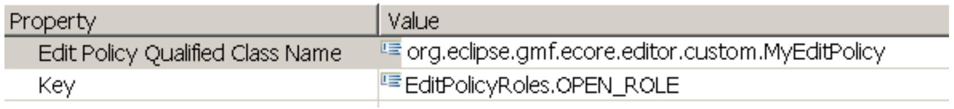
Property	Value
Edit Policy Qualified Class Name	org.eclipse.gmf.ecore.editor.custom.MyEditPolicy
Key	EditPolicyRoles.OPEN_ROLE

Customization – Manual classes

- Example


Gen Child Label Node EAttributeEditPart
Metamodel Type EAttribute
Figure Viewmap org.eclipse.gmf.runti
Type Model Facet
Feature Label Model Facet
New Child
New Sibling
Undo Delete
Parent Assigned Viewmap
Custom Behaviour
Shared Behaviour
Open Diagram Behaviour

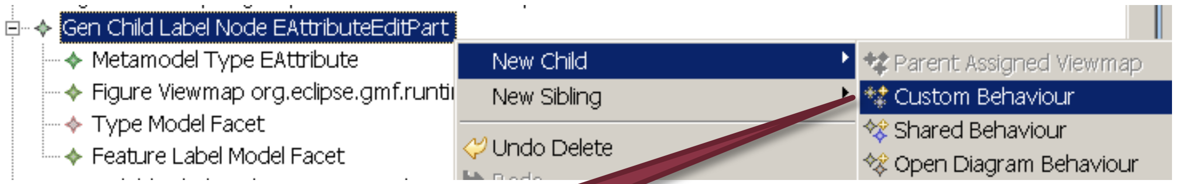
Custom behaviour setting



Property	Value
Edit Policy Qualified Class Name	org.eclipse.gmf.ecore.editor.custom.MyEditPolicy
Key	EditPolicyRoles.OPEN_ROLE

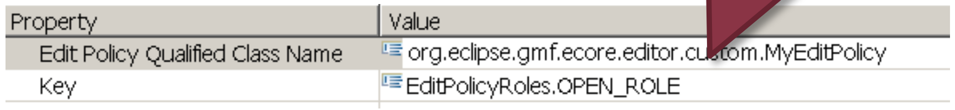
Customization – Manual classes

- Example


Gen Child Label Node EAttributeEditPart
Metamodel Type EAttribute
Figure Viewmap org.eclipse.gmf.runti
Type Model Facet
Feature Label Model Facet
New Child
New Sibling
Undo Delete
Parent Assigned Viewmap
Custom Behaviour
Shared Behaviour
Open Diagram Behaviour

Custom behaviour setting

Implementation class

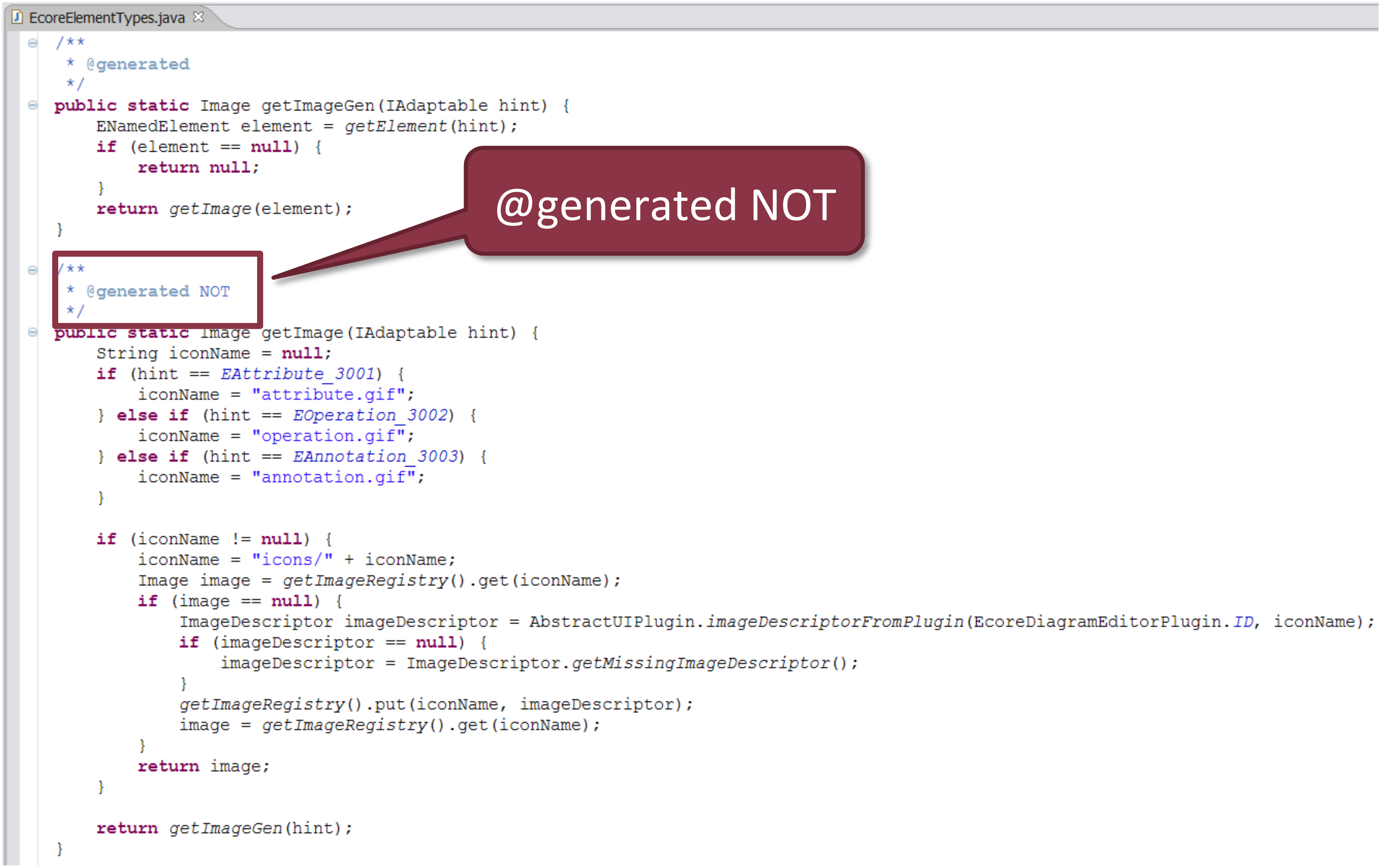


Property	Value
Edit Policy Qualified Class Name	org.eclipse.gmf.ecore.editor.custom.MyEditPolicy
Key	EditPolicyRoles.OPEN_ROLE

Customization – Code Overwrite

- Overwriting generated methods
- @generated NOT
  - Like in case of EMF
  - Not required for new methods
- Advantages
  - Simple
- Disadvantages
  - Brittle

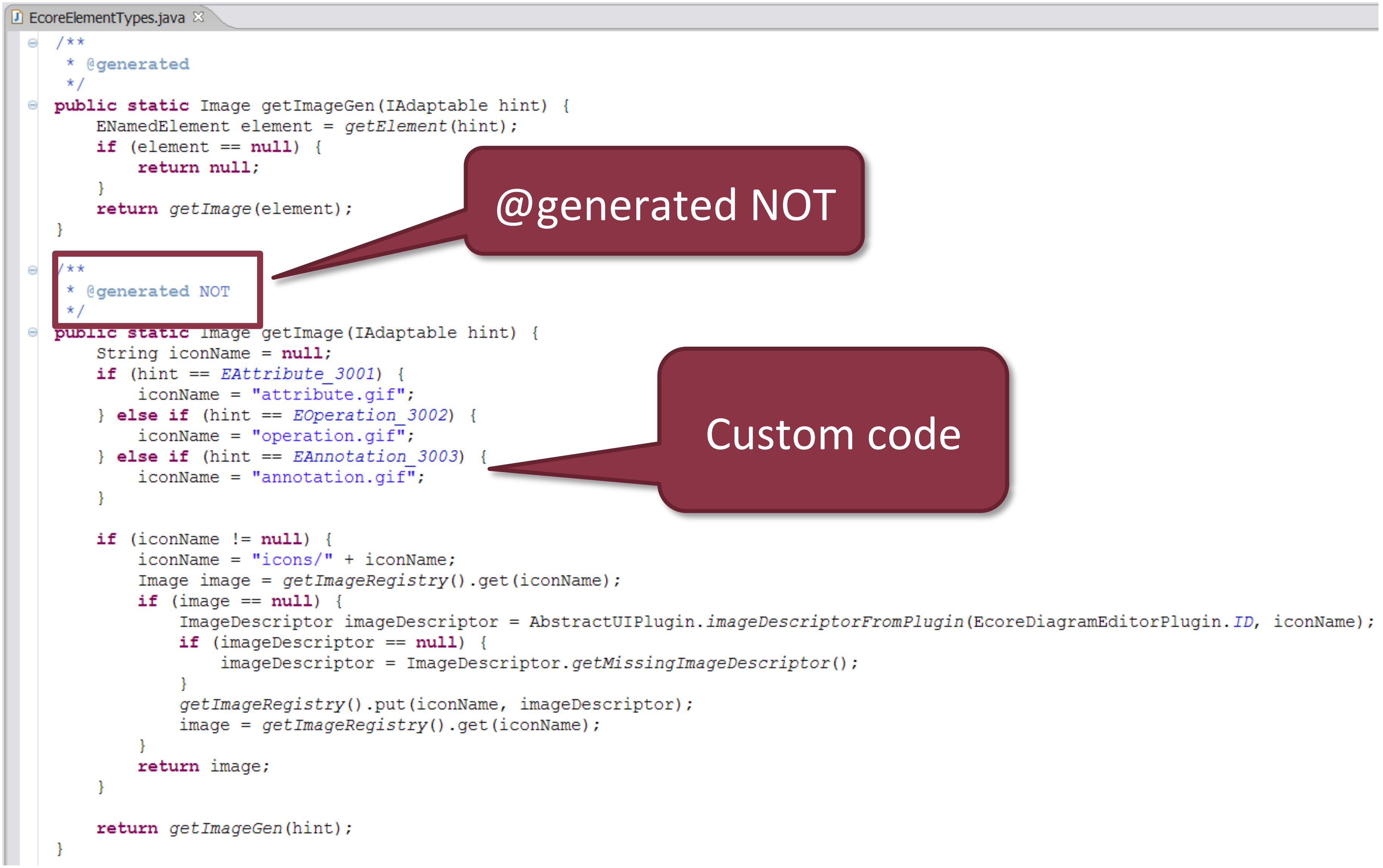
Customization – Code Overwrite



```
/**
 * @generated
 */
public static Image getImageGen(IAdaptable hint) {
    ENamedElement element = getElement(hint);
    if (element == null) {
        return null;
    }
    return getImage(element);
}

/**
 * @generated NOT
 */
public static Image getImage(IAdaptable hint) {
    String iconName = null;
    if (hint == EAttribute_3001) {
        iconName = "attribute.gif";
    } else if (hint == EOperation_3002) {
        iconName = "operation.gif";
    } else if (hint == EAnnotation_3003) {
        iconName = "annotation.gif";
    }

    if (iconName != null) {
        iconName = "icons/" + iconName;
        Image image = getImageRegistry().get(iconName);
        if (image == null) {
            ImageDescriptor imageDescriptor = AbstractUIPlugin.imageDescriptorFromPlugin(EcoreDiagramEditorPlugin.ID, iconName);
            if (imageDescriptor == null) {
                imageDescriptor = ImageDescriptor.getMissingImageDescriptor();
            }
            getImageRegistry().put(iconName, imageDescriptor);
            image = getImageRegistry().get(iconName);
        }
        return image;
    }

    return getImageGen(hint);
}
```

@generated NOT

Customization – Code Overwrite



```java
/**
 * @generated
 */
public static Image getImageGen(IAdaptable hint) {
    ENamedElement element = getElement(hint);
    if (element == null) {
        return null;
    }
    return getImage(element);
}

/**
 * @generated NOT
 */
public static Image getImage(IAdaptable hint) {
    String iconName = null;
    if (hint == EAttribute_3001) {
        iconName = "attribute.gif";
    } else if (hint == EOperation_3002) {
        iconName = "operation.gif";
    } else if (hint == EAnnotation_3003) {
        iconName = "annotation.gif";
    }

    if (iconName != null) {
        iconName = "icons/" + iconName;
        Image image = getImageRegistry().get(iconName);
        if (image == null) {
            ImageDescriptor imageDescriptor = AbstractUIPlugin.imageDescriptorFromPlugin(EcoreDiagramEditorPlugin.ID, iconName);
            if (imageDescriptor == null) {
                imageDescriptor = ImageDescriptor.getMissingImageDescriptor();
            }
            getImageRegistry().put(iconName, imageDescriptor);
            image = getImageRegistry().get(iconName);
        }
        return image;
    }

    return getImageGen(hint);
}
```

Customization – Template Extensions

- JET/Xpand template updates
  - Template directory setting in genmodel
- Advantage
  - Reusable
- Disadvantages
  - JET/Xpand **and** GMF runtime knowledge needed
  - Template update is non-trivial

Customization – Template extension

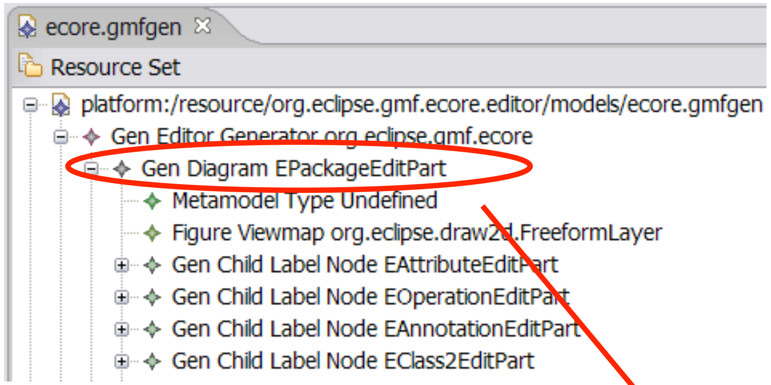
- Example

```
«AROUND getAdaptableImage FOR gmfgen::GenDiagram-»
    «EXPAND xpt::Common::generatedMemberComment»
public static org.eclipse.swt.graphics.Image getImage(org.eclipse.core.runtime.IAdaptable hint) {
    org.eclipse.gmf.runtime.emf.type.core.IElementType elementType = (org.eclipse.gmf.runtime.emf.type.core.IElementType)
        hint.getAdapter(org.eclipse.gmf.runtime.emf.type.core.IElementType.class);
    «EXPAND addCustomIcon FOREACH palette.groups.entries-»
    «EXPAND xpt::diagram::providers::ElementTypes::getNamedElement-»
    return getImage(element);
}
«ENDAROUND»

«DEFINE addCustomIcon FOR gmfgen::ToolEntry-»
    «IF null != largeIconPath-»
        «EXPAND getImage(this) FOREACH genNodes-»
    «ENDIF-»
«ENDDEFINE»

«DEFINE getImage(gmfgen::ToolEntry entry) FOR gmfgen::GenNode-»
if (elementType == «getUniqueIdentifier()») {
    String key = "«entry.largeIconPath»";
    org.eclipse.swt.graphics.Image image = getImageRegistry().get(key);
    if (image == null) {
        org.eclipse.jface.resource.ImageDescriptor imageDescriptor = org.eclipse.ui.plugin.AbstractUIPlugin.
            imageDescriptorFromPlugin(«getDiagram().editorGen.plugin.getActivatorQualifiedClassName()».ID, key);
        if (imageDescriptor == null) {
            imageDescriptor = org.eclipse.jface.resource.ImageDescriptor.getMissingImageDescriptor();
        }
        getImageRegistry().put(key, imageDescriptor);
        image = getImageRegistry().get(key);
    }
    return image;
}
«ENDDEFINE»
```

Xpand template

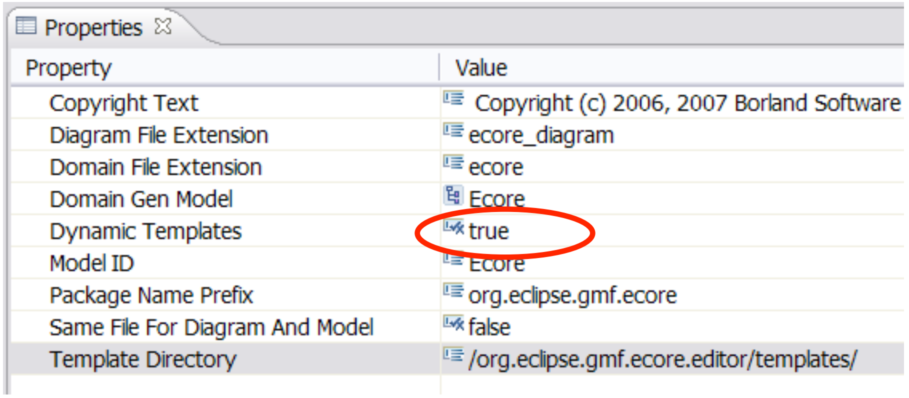
Customization – Template extension

- Example



ecore.gmfgen
Resource Set

```
platform:/resource/org.eclipse.gmf.ecore.editor/models/ecore.gmfgen
  Gen Editor Generator org.eclipse.gmf.ecore
    Gen Diagram EPackageEditPart
      Metamodel Type Undefined
      Figure Viewmap org.eclipse.draw2d.FreeformLayer
      Gen Child Label Node EAttributeEditPart
      Gen Child Label Node EOperationEditPart
      Gen Child Label Node EAnnotationEditPart
      Gen Child Label Node EClass2EditPart
```



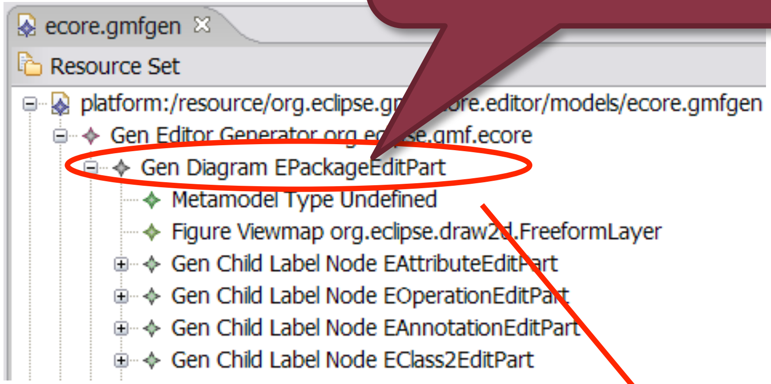
Properties

Property	Value
Copyright Text	Copyright (c) 2006, 2007 Borland Software
Diagram File Extension	ecore_diagram
Domain File Extension	ecore
Domain Gen Model	Ecore
Dynamic Templates	true
Model ID	Ecore
Package Name Prefix	org.eclipse.gmf.ecore
Same File For Diagram And Model	false
Template Directory	/org.eclipse.gmf.ecore.editor/templates/

Customization – Template extension

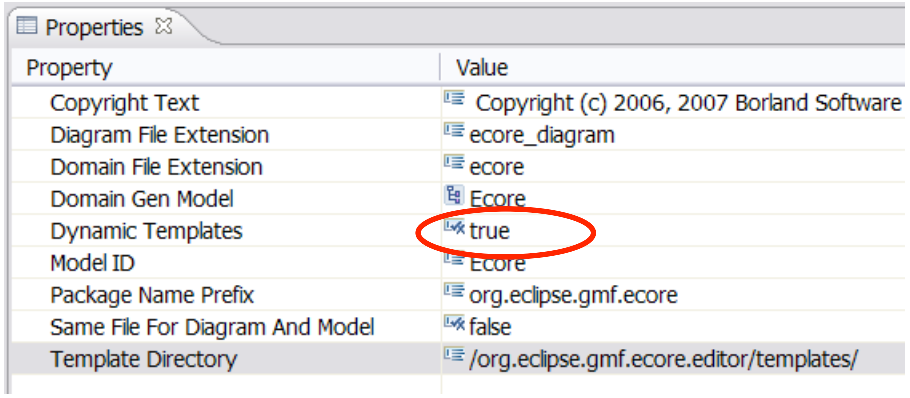
- Example

Select model element


ecore.gmfgen

Resource Set

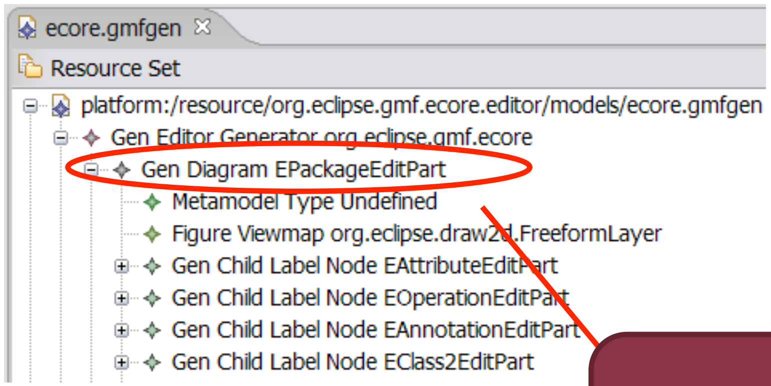
- platform:/resource/org.eclipse.gmf.ecore.editor/models/ecore.gmfgen
  - Gen Editor Generator org.eclipse.gmf.ecore
    - Gen Diagram EPackageEditPart
      - Metamodel Type Undefined
      - Figure Viewmap org.eclipse.draw2d.FreeformLayer
      - Gen Child Label Node EAttributeEditPart
      - Gen Child Label Node EOperationEditPart
      - Gen Child Label Node EAnnotationEditPart
      - Gen Child Label Node EClass2EditPart


Properties

Property	Value
Copyright Text	Copyright (c) 2006, 2007 Borland Software
Diagram File Extension	ecore_diagram
Domain File Extension	ecore
Domain Gen Model	Ecore
Dynamic Templates	true
Model ID	Ecore
Package Name Prefix	org.eclipse.gmf.ecore
Same File For Diagram And Model	false
Template Directory	/org.eclipse.gmf.ecore.editor/templates/

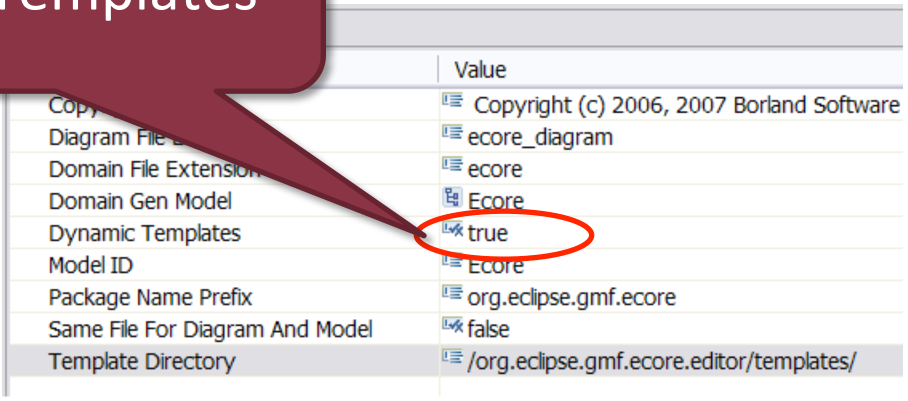
Customization – Template extension

- Example



ecore.gmfgen
Resource Set
- platform:/resource/org.eclipse.gmf.ecore.editor/models/ecore.gmfgen
  - Gen Editor Generator org.eclipse.gmf.ecore
    - Gen Diagram EPackageEditPart
      - Metamodel Type Undefined
      - Figure Viewmap org.eclipse.draw2d.FreeformLayer
      - Gen Child Label Node EAttributeEditPart
      - Gen Child Label Node EOperationEditPart
      - Gen Child Label Node EAnnotationEditPart
      - Gen Child Label Node EClass2EditPart

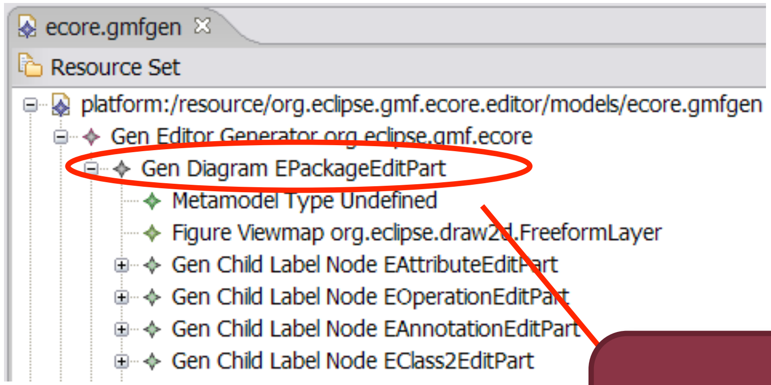
Dynamic Templates



	Value
Copy	Copyright (c) 2006, 2007 Borland Software
Diagram File	ecore_diagram
Domain File Extension	ecore
Domain Gen Model	Ecore
Dynamic Templates	true
Model ID	Ecore
Package Name Prefix	org.eclipse.gmf.ecore
Same File For Diagram And Model	false
Template Directory	/org.eclipse.gmf.ecore.editor/templates/

Customization – Template extension

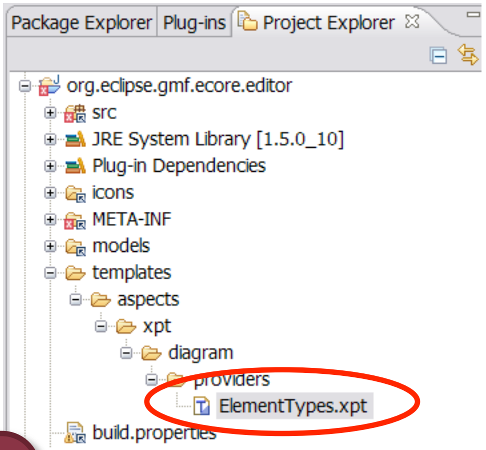
- Example



ecore.gmfgen

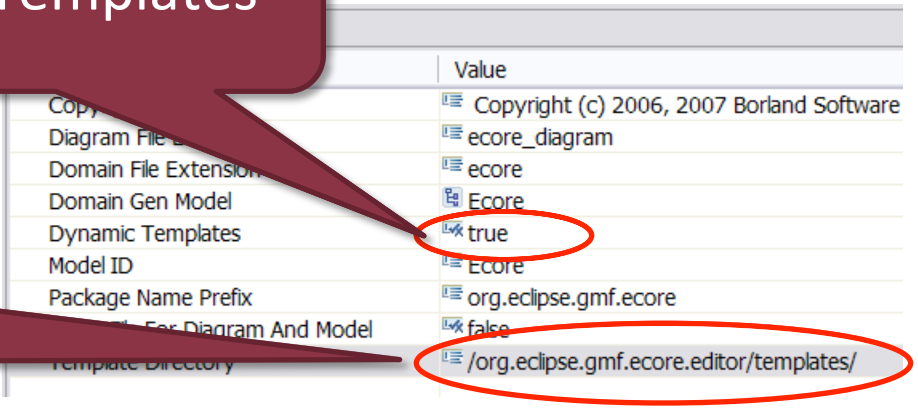
Resource Set

- platform:/resource/org.eclipse.gmf.ecore.editor/models/ecore.gmfgen
  - Gen Editor Generator org.eclipse.gmf.ecore
    - Gen Diagram EPackageEditPart
      - Metamodel Type Undefined
      - Figure Viewmap org.eclipse.draw2d.FreeformLayer
      - Gen Child Label Node EAttributeEditPart
      - Gen Child Label Node EOperationEditPart
      - Gen Child Label Node EAnnotationEditPart
      - Gen Child Label Node EClass2EditPart



Package Explorer | Plug-ins | Project Explorer

- org.eclipse.gmf.ecore.editor
  - src
  - JRE System Library [1.5.0_10]
  - Plug-in Dependencies
  - icons
  - META-INF
  - models
  - templates
    - aspects
      - xpt
        - diagram
          - providers
            - ElementTypes.xpt
  - build.properties



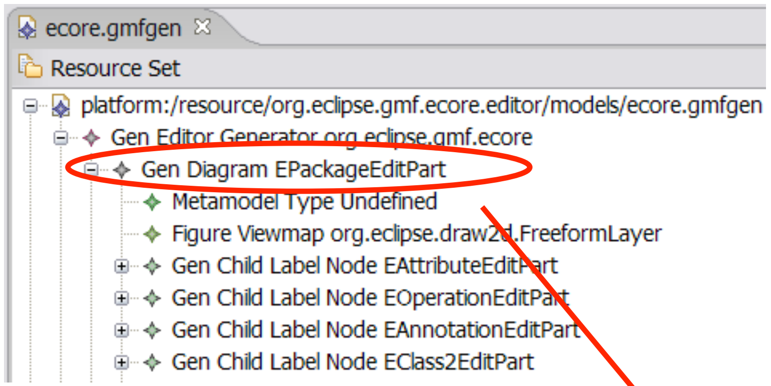
	Value
Copy	Copyright (c) 2006, 2007 Borland Software
Diagram File	ecore_diagram
Domain File Extension	ecore
Domain Gen Model	Ecore
Dynamic Templates	true
Model ID	Ecore
Package Name Prefix	org.eclipse.gmf.ecore
For Diagram And Model	false
Template Directory	/org.eclipse.gmf.ecore.editor/templates/

Dynamic Templates

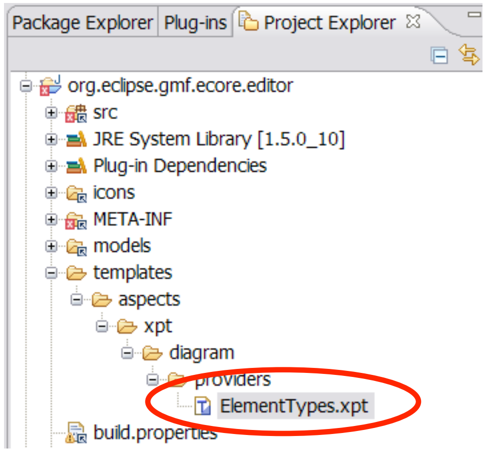
Reference to custom template

Customization – Template extension

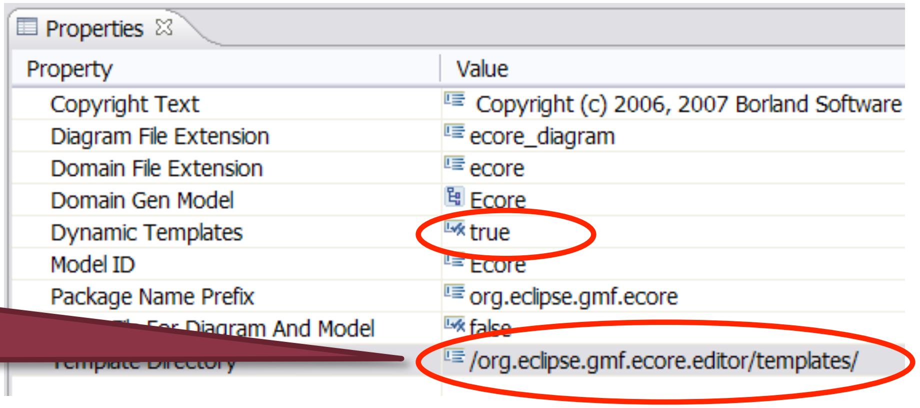
- Example



ecore.gmfgen
Resource Set
platform:/resource/org.eclipse.gmf.ecore.editor/models/ecore.gmfgen
Gen Editor Generator org.eclipse.gmf.ecore
Gen Diagram EPackageEditPart
Metamodel Type Undefined
Figure Viewmap org.eclipse.draw2d.FreeformLayer
Gen Child Label Node EAttributeEditPart
Gen Child Label Node EOperationEditPart
Gen Child Label Node EAnnotationEditPart
Gen Child Label Node EClass2EditPart



Package Explorer | Plug-ins | Project Explorer
org.eclipse.gmf.ecore.editor
src
JRE System Library [1.5.0_10]
Plug-in Dependencies
icons
META-INF
models
templates
aspects
xpt
diagram
providers
ElementTypes.xpt
build.properties



Properties

Property	Value
Copyright Text	Copyright (c) 2006, 2007 Borland Software
Diagram File Extension	ecore_diagram
Domain File Extension	ecore
Domain Gen Model	Ecore
Dynamic Templates	true
Model ID	Ecore
Package Name Prefix	org.eclipse.gmf.ecore
Same File For Diagram And Model	false
Template Directory	/org.eclipse.gmf.ecore.editor/templates/

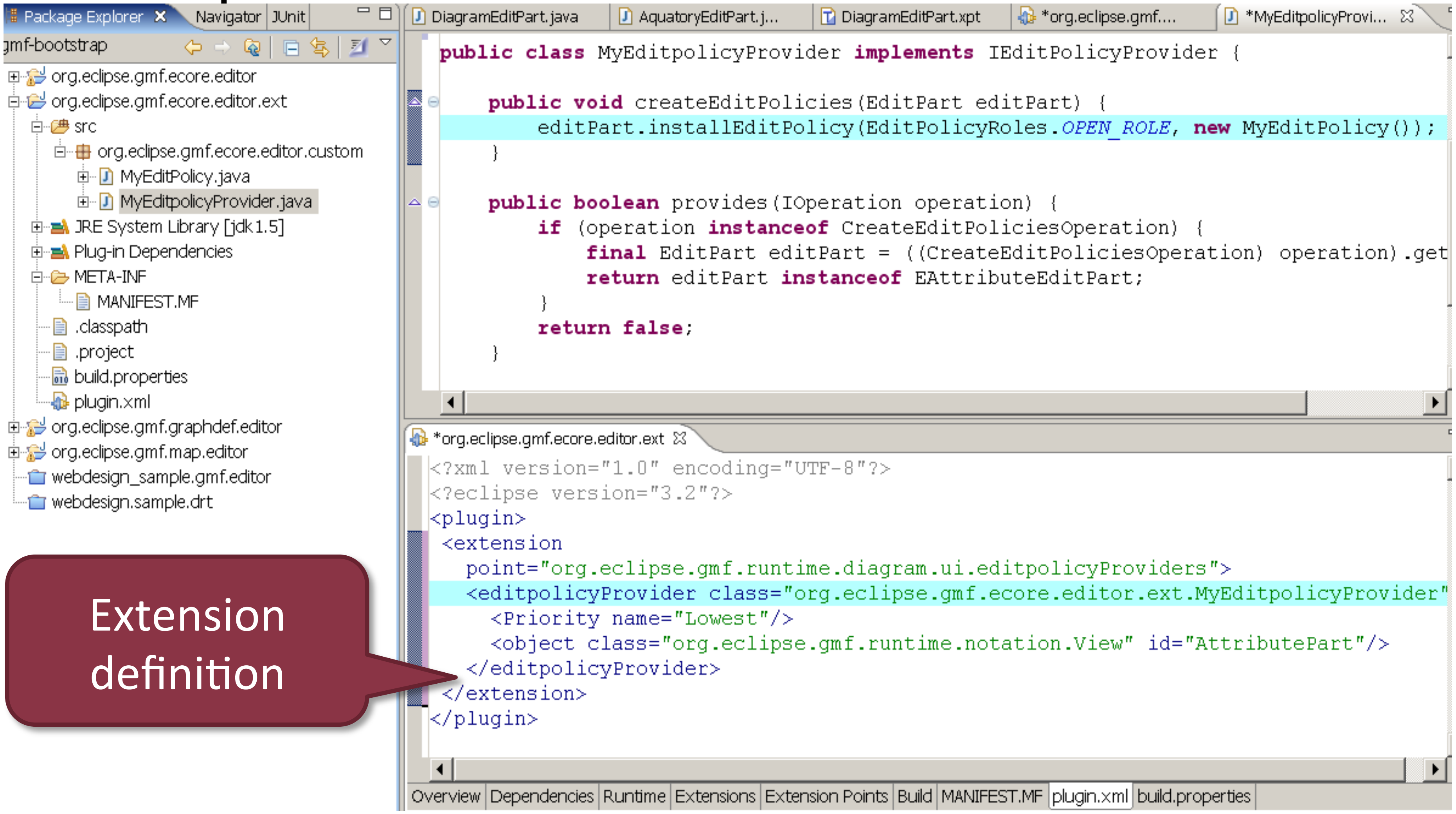
Reference to custom template

Customization – Extension point

- *Provider extension points
  - View, EditPart, EditPolicy, Icon stb.
- Advantage
  - Safe
- Disadvantage
  - Boilerplate code needed

Customization – Extension point

- Example



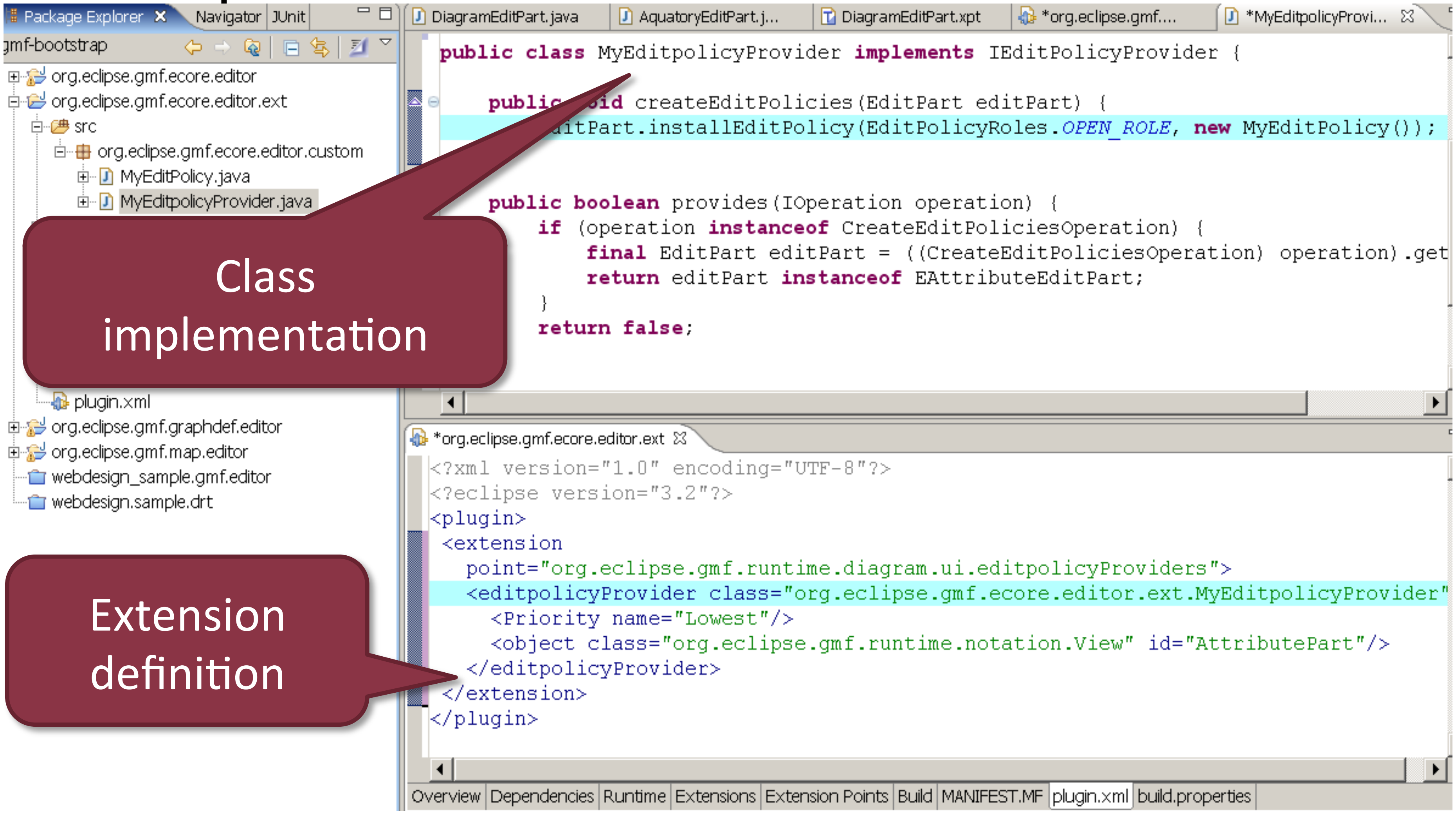
```java
public class MyEditpolicyProvider implements IEditPolicyProvider {

    public void createEditPolicies(EditPart editPart) {
        editPart.installEditPolicy(EditPolicyRoles.OPEN_ROLE, new MyEditPolicy());
    }

    public boolean provides(IOperation operation) {
        if (operation instanceof CreateEditPoliciesOperation) {
            final EditPart editPart = ((CreateEditPoliciesOperation) operation).get
            return editPart instanceof EAttributeEditPart;
        }
        return false;
    }
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<?eclipse version="3.2"?>
<plugin>
 <extension
   point="org.eclipse.gmf.runtime.diagram.ui.editpolicyProviders">
   <editpolicyProvider class="org.eclipse.gmf.ecore.editor.ext.MyEditpolicyProvider"
     <Priority name="Lowest"/>
     <object class="org.eclipse.gmf.runtime.notation.View" id="AttributePart"/>
   </editpolicyProvider>
 </extension>
</plugin>
```

Extension definition

Customization – Extension point

- Example



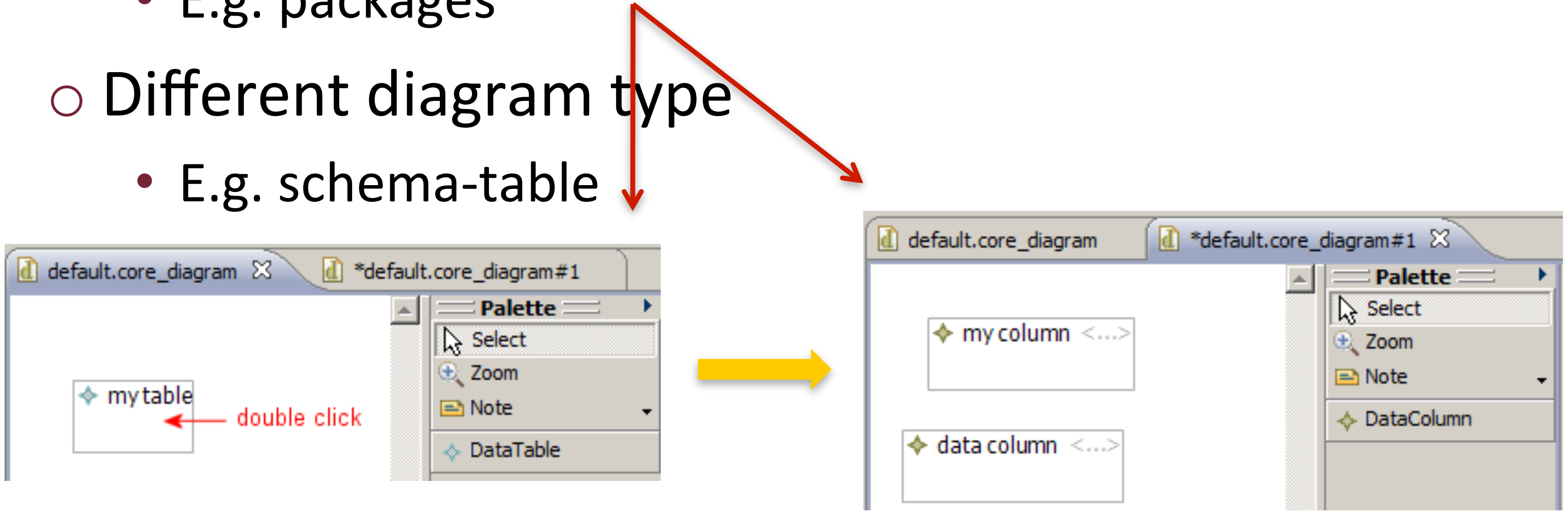
```java
public class MyEditpolicyProvider implements IEditPolicyProvider {

    public void createEditPolicies(EditPart editPart) {
        editPart.installEditPolicy(EditPolicyRoles.OPEN_ROLE, new MyEditPolicy());


    public boolean provides(IOperation operation) {
        if (operation instanceof CreateEditPoliciesOperation) {
            final EditPart editPart = ((CreateEditPoliciesOperation) operation).get
            return editPart instanceof EAttributeEditPart;
        }
        return false;
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<?eclipse version="3.2"?>
<plugin>
 <extension
   point="org.eclipse.gmf.runtime.diagram.ui.editpolicyProviders">
   <editpolicyProvider class="org.eclipse.gmf.ecore.editor.ext.MyEditpolicyProvider"
     <Priority name="Lowest"/>
     <object class="org.eclipse.gmf.runtime.notation.View" id="AttributePart"/>
   </editpolicyProvider>
 </extension>
</plugin>
```

Additional GMF techniques

Diagram partitioning

- Dig in/Drill down
- Two cases:
  - Same diagram type (recursive containment)
    - E.g. packages
  - Different diagram type
    - E.g. schema-table



Initial property values

- EMF model
  - Default value property
  - Simple values
- GMF Mapping model
  - Node/Link Mapping
    - Feature Seq Initializer
      - Feature Value Spec
        - Value Expression
  - OCL expression

Validation constraints - Connection

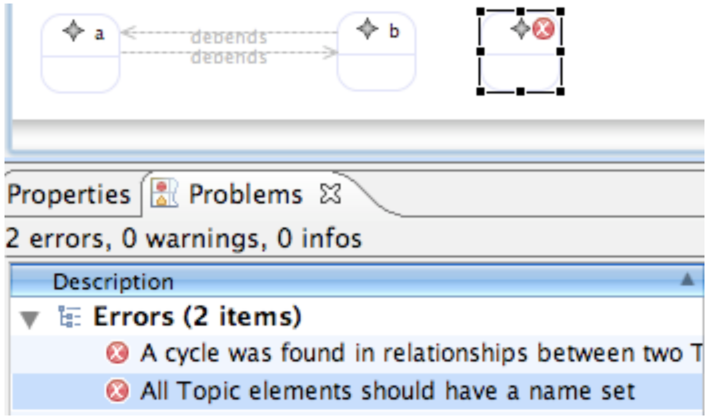
- In OCL
- Mapping Model
  - Link Mapping
    - Link Constraint
- E.g. no self-loops possible
  - self <> oppositeEnd
- Constraints enforced at element creation!

Constraints

- OCL language as well
- Mapping model
  - Audit Container
    - Audit Rule
      - Target
      - Constraint
- Only manual validation

Validation

- Enablement: Generator model
  - Gen Diagram
    - Validation Enabled
    - Validation Decorators
    - Live Validation UI Feedback
- Start validation: Edit/Validate

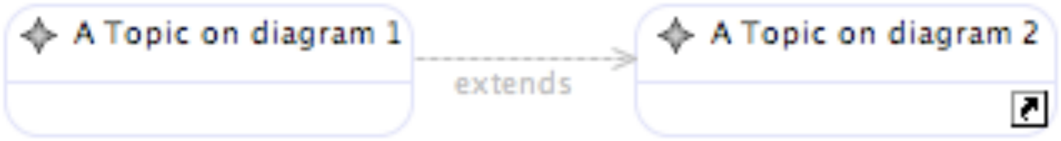


2 errors, 0 warnings, 0 infos

Errors (2 items)
A cycle was found in relationships between two T
All Topic elements should have a name set

Shortcut

- Live copies
  - Reference in another diagram
- Setup options: Generator Model
  - Gen Diagram
    - Source: Shortcuts Provided For = target diagram extension
    - Target: Contains Shortcuts To = source diagram extension



Summary

Summary – Graphical Editors

	GEF	Graphiti	GMF
Model	Any	EMF	EMF
Non-graph display	Possible	Not	Many, complex coding needed
Implementation amount	Many, repetitive code	Medium amount of code	Mostly modeling, some coding
Workflow	Only coding	Only coding	Multi-step

Graphiti, GMF – When to use

- EMF model
- Display is basically a graph
- Quick implementation

Graphiti or GMF?

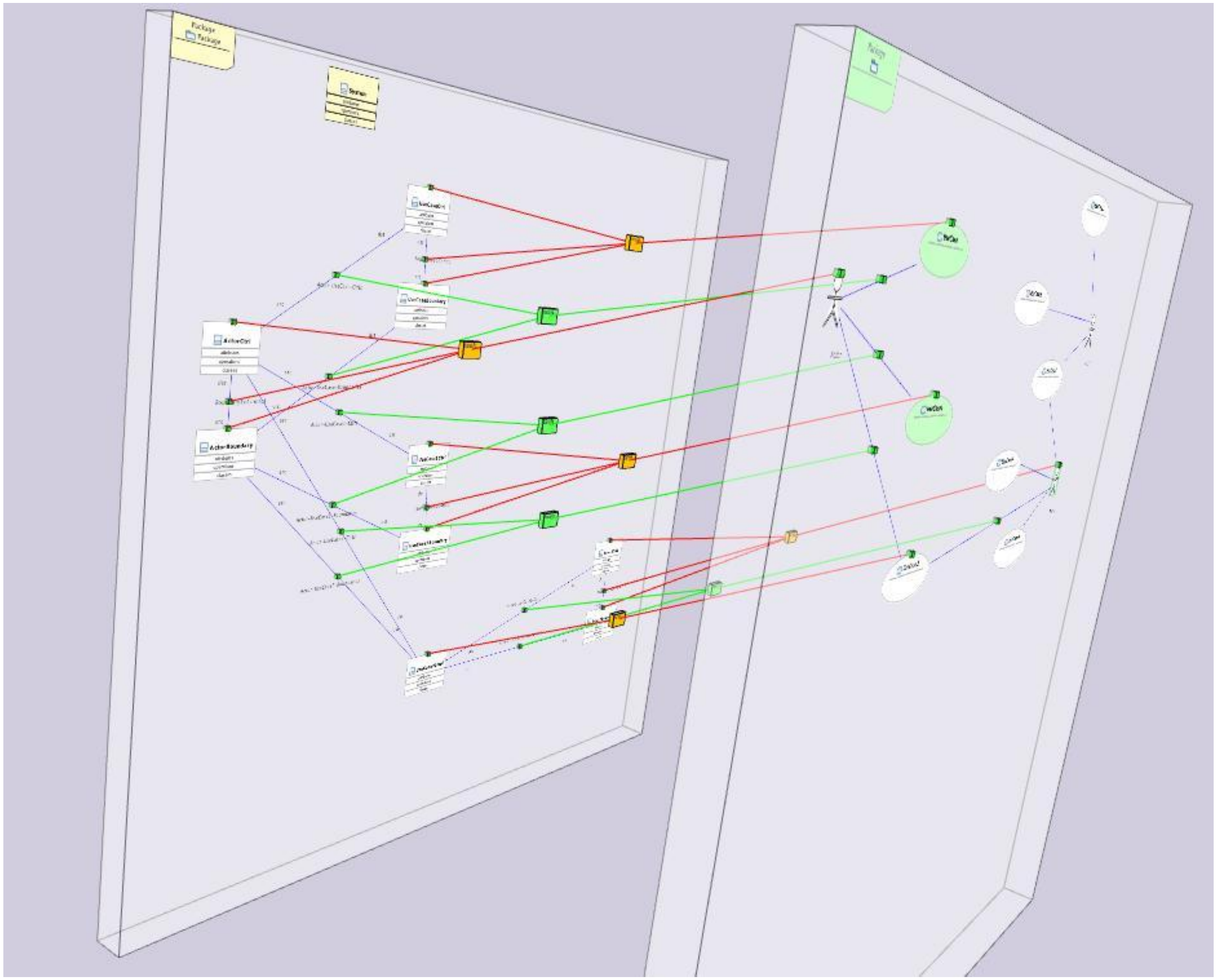
- Graphiti
  - Simpler cases
  - Easier to manage model updates
- GMF
  - Quick prototyping
    - Primitive concrete syntax
  - More complex cases
    - Model remains unchanged
    - Advanced features

Further technologies

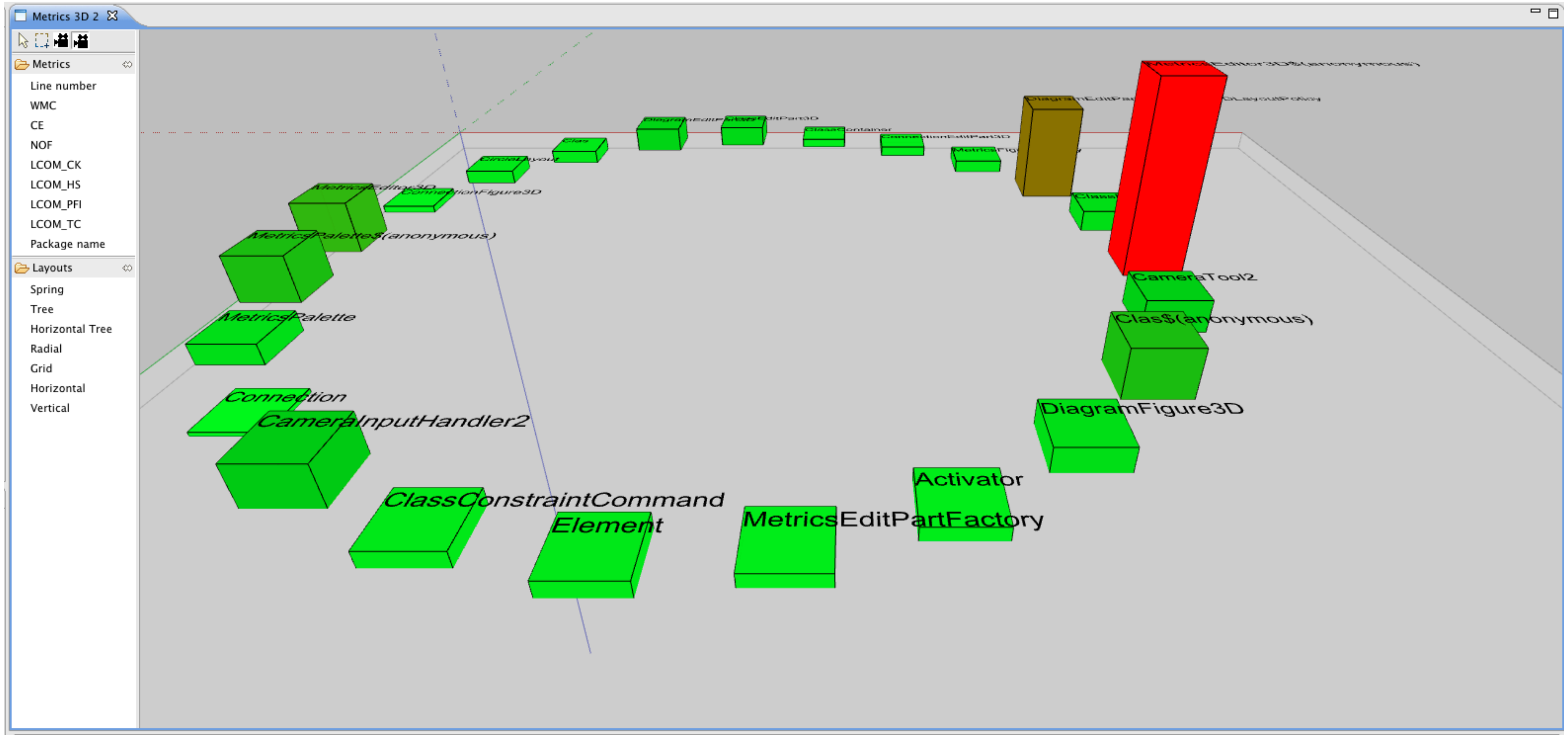
GEF3D

- GEF extension with 3D support
- Extending
  - Draw2D with 3D drawing primitives
  - GEF EditParts with 3D EditPolicies
- Supports
  - Existing editors on a 2D plane (dubbed 2.5D)
  - Entirely 3D editors

GEF3D – 2.5D Example



GEF3D – Full 3D Example



Zest Graph Layout Library

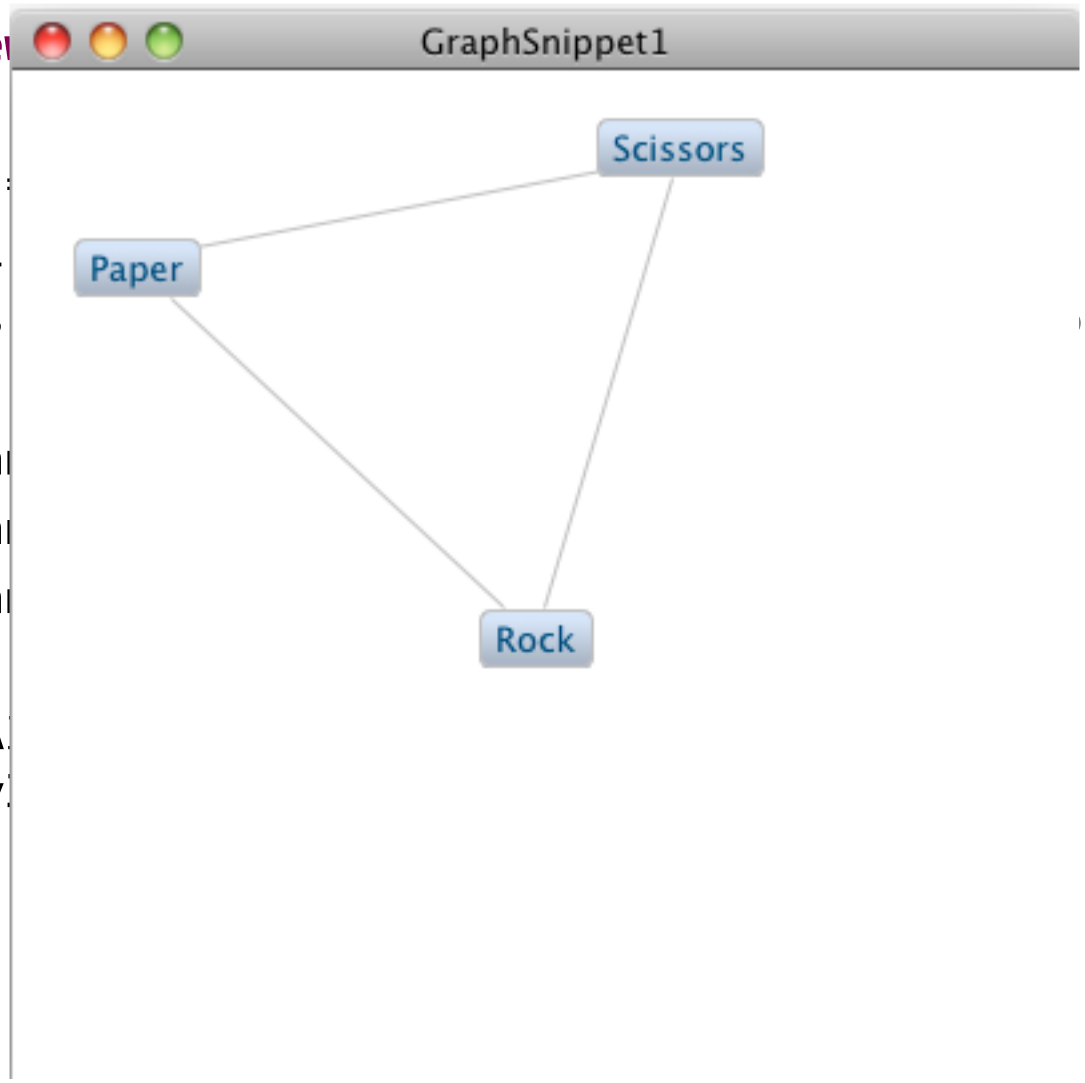
- Graph widgets created in Draw2D, supports
  - Automatic layouting
  - Custom figure implementation
  - High level API similar to JFace Viewers
- Layout algorithms reusable in GEF-based editors
  - Requires coding custom GEF layout based on Zest

Zest example

```
Graph g = new Graph(shell, SWT.NONE);

GraphNode n = new GraphNode(g, SWT.NONE, "Paper");
GraphNode n2 = new GraphNode(g, SWT.NONE, "Rock");
GraphNode n3 = new GraphNode(g, SWT.NONE, "Scissors");

new GraphConnection(g, SWT.NONE, n, n2);
new GraphConnection(g, SWT.NONE, n2, n3);
new GraphConnection(g, SWT.NONE, n3, n);

g.setLayoutAlgorithm(new SpringLayoutAlgorithm
  (LayoutStyles.NO_LAYOUT_NODE_RESIZING), true);
```

Zest example

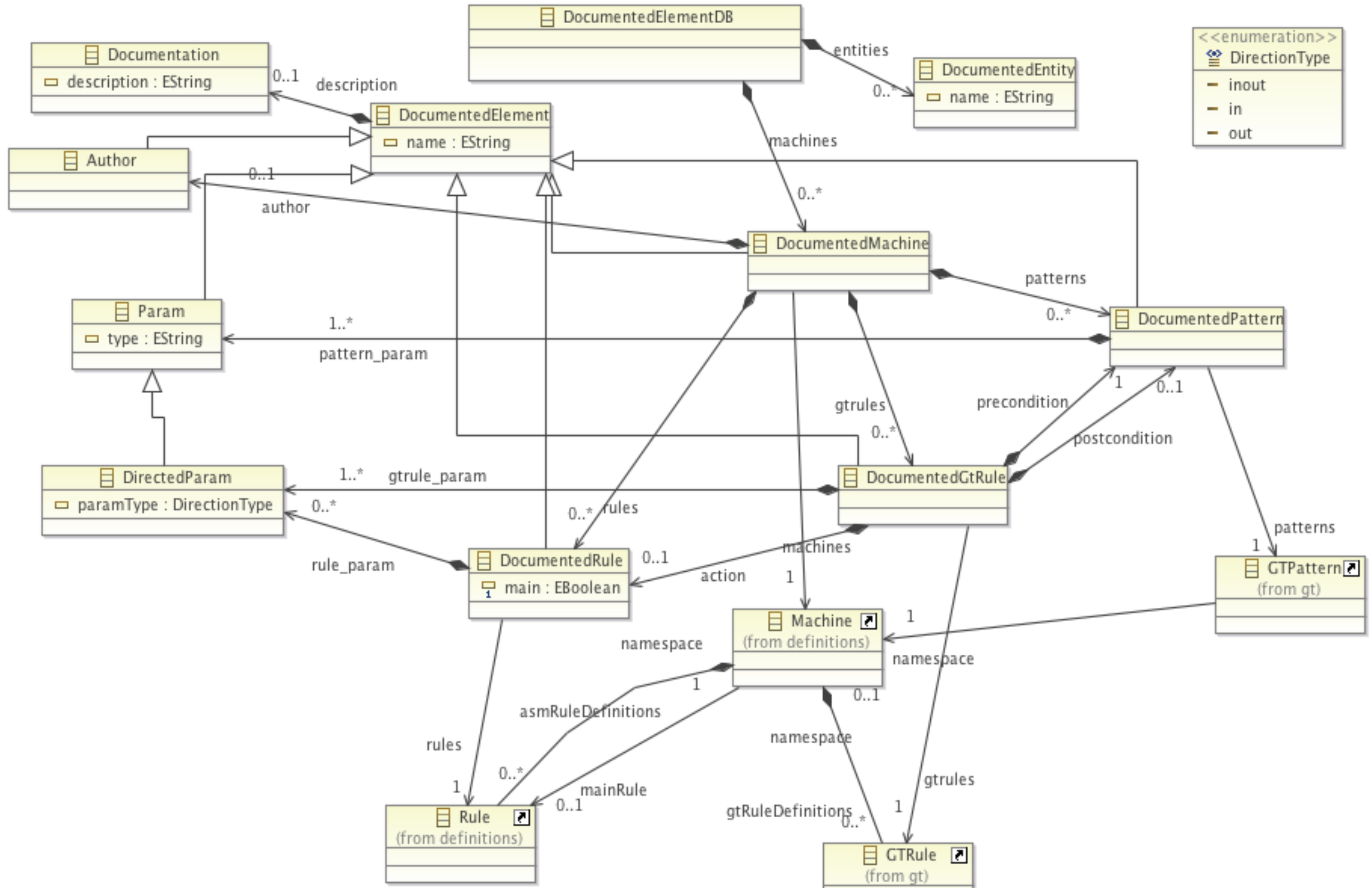
```
Graph g = ne

GraphNode n
GraphNode n2
GraphNode n3                                              ;

new GraphCon
new GraphCon
new GraphCon

g.setLayoutA
  (LayoutSty
```


GraphSnippet1

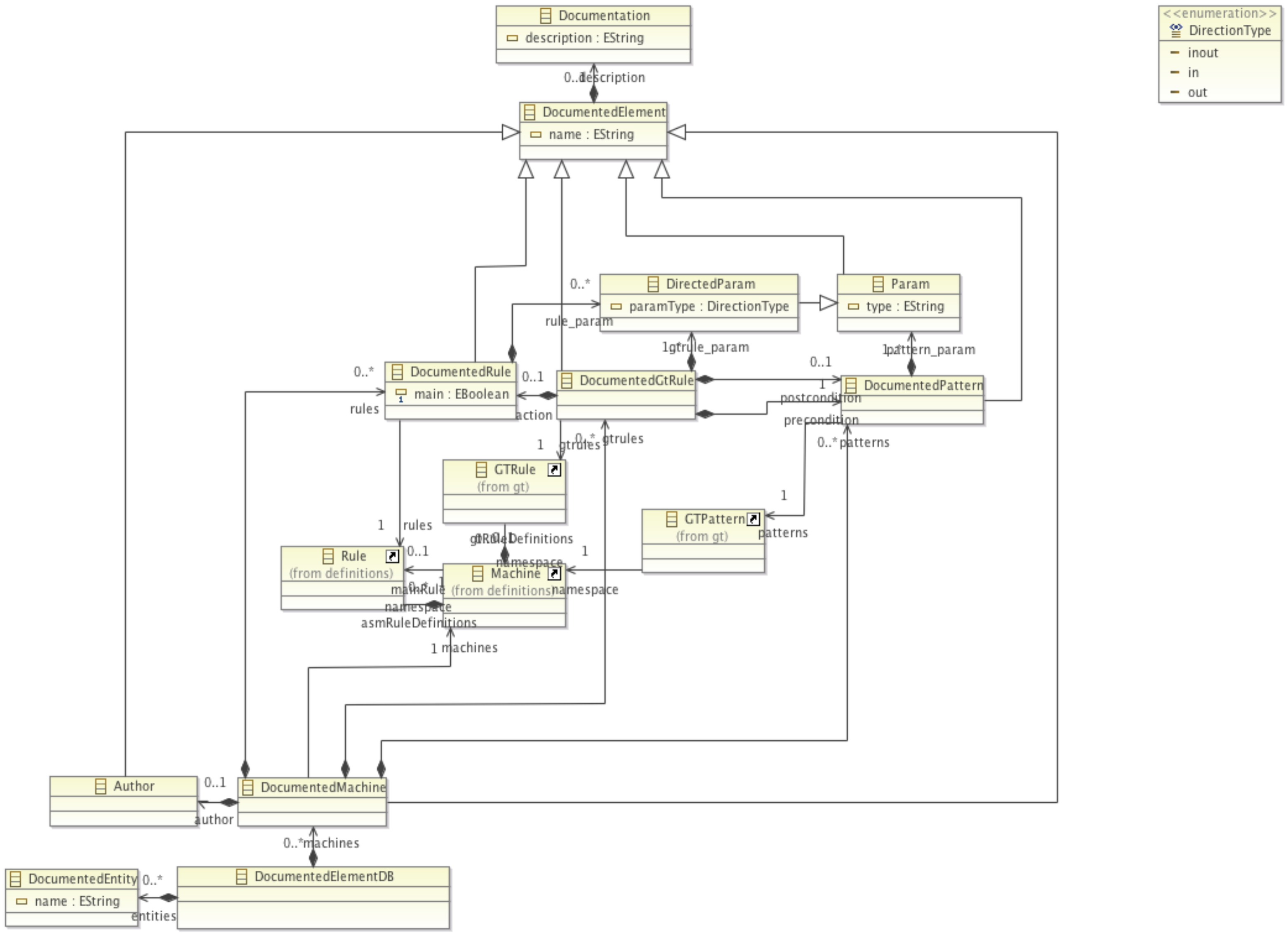
KIELER project

- Academic project related to automatic layouting
  - Automatic integration to GEF/GMF/Graphiti editors
  - Good defaults for class-diagram like structures

KIELER Example - Before



KIELER Example - After



Spray

- Textual syntax for
  - Defining and integrating Graphiti pictograms
- Conceptually similar to GMF Tooling
- But:
  - Some technological issues

Sirius

- New eclipse.org project
  - Formerly Obeo Designer
- High-level support for model editors
  - Re-uses GMF
  - Supports multiple viewpoints

EuGENia

- Part of the Epsilon (Model Transformation)
  - Generates GMF tooling models
  - Much easier to understand
  - Limited capabilities (wrt. to GMF Tooling)