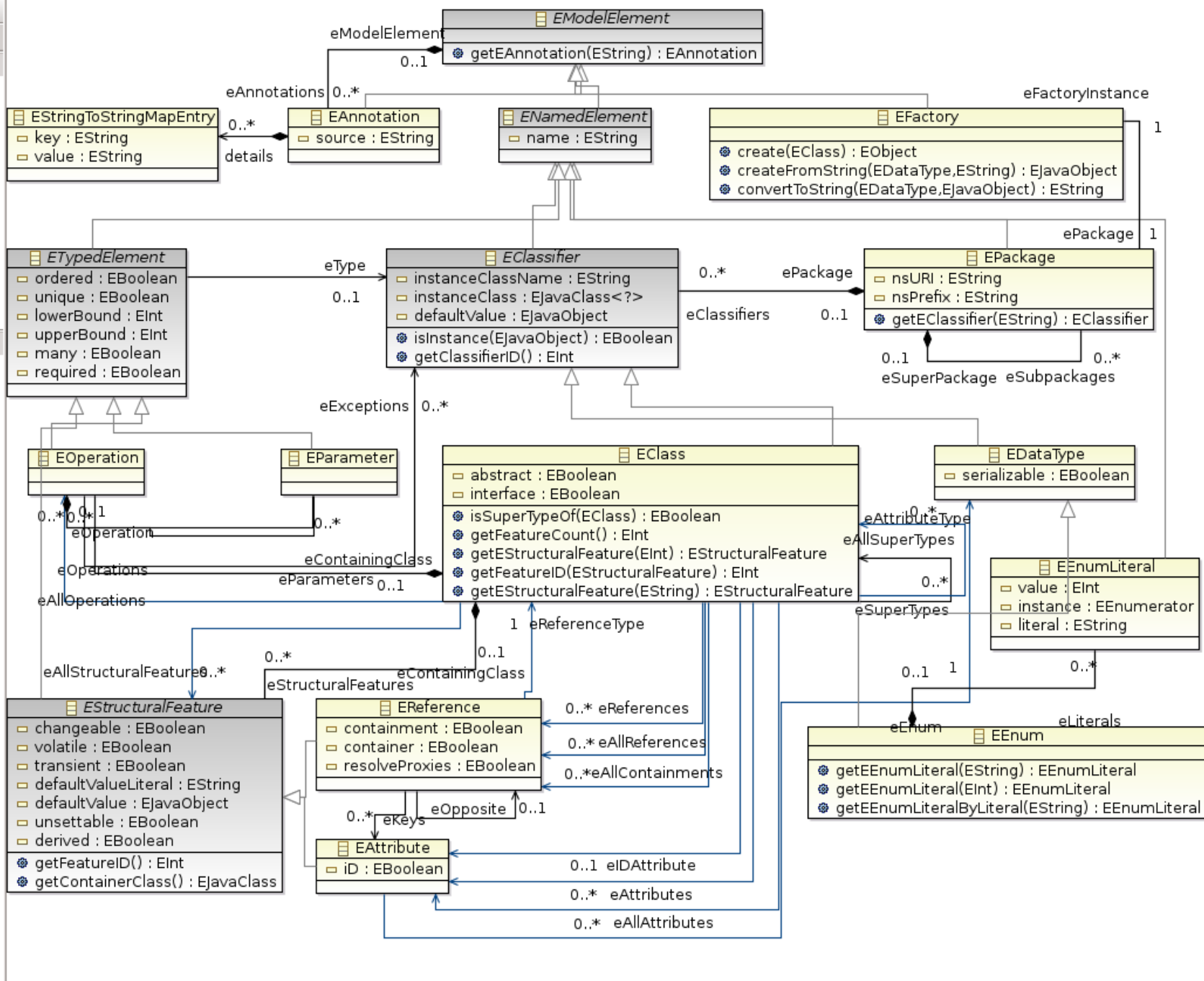


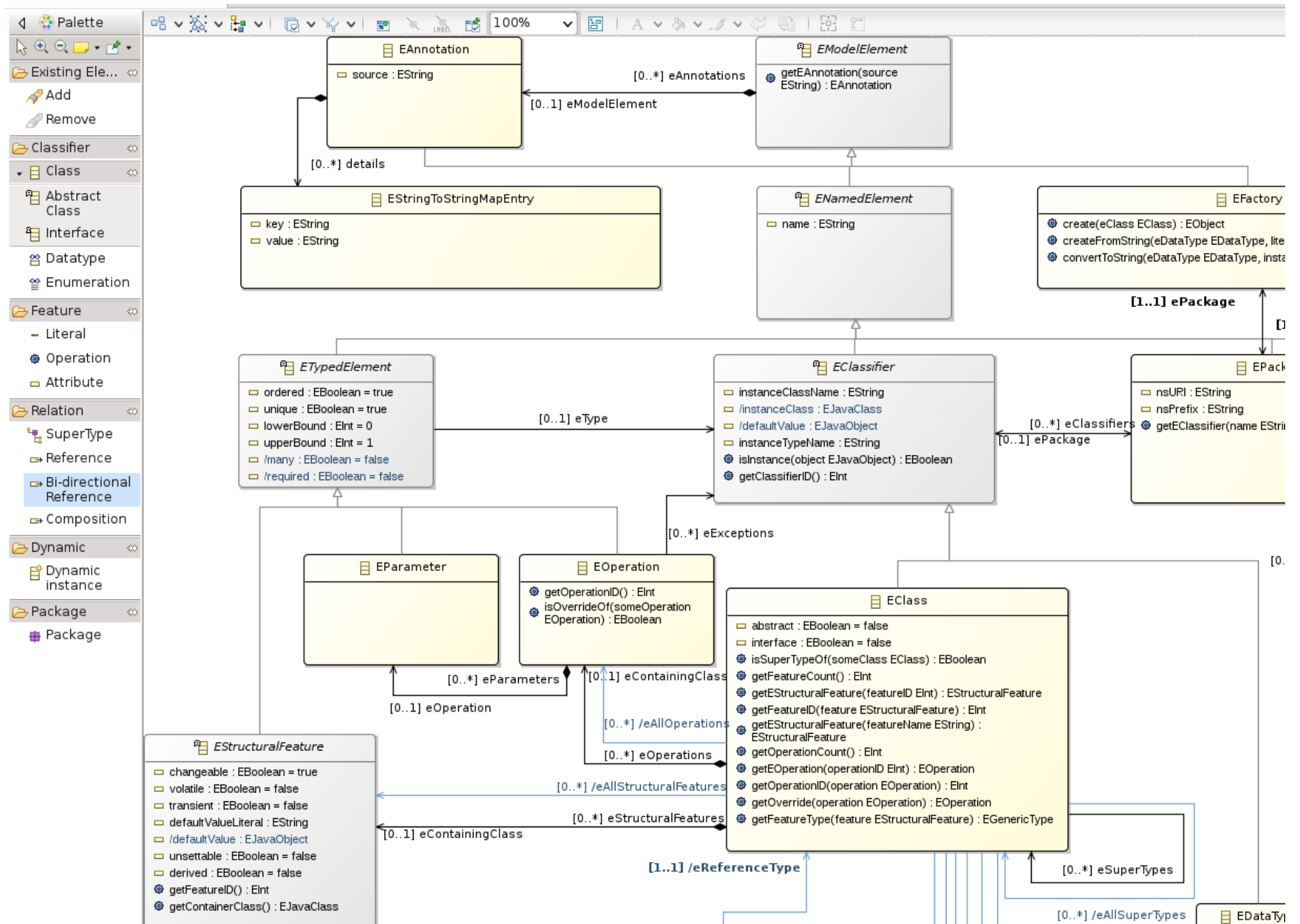


# EcoreTools 2.0

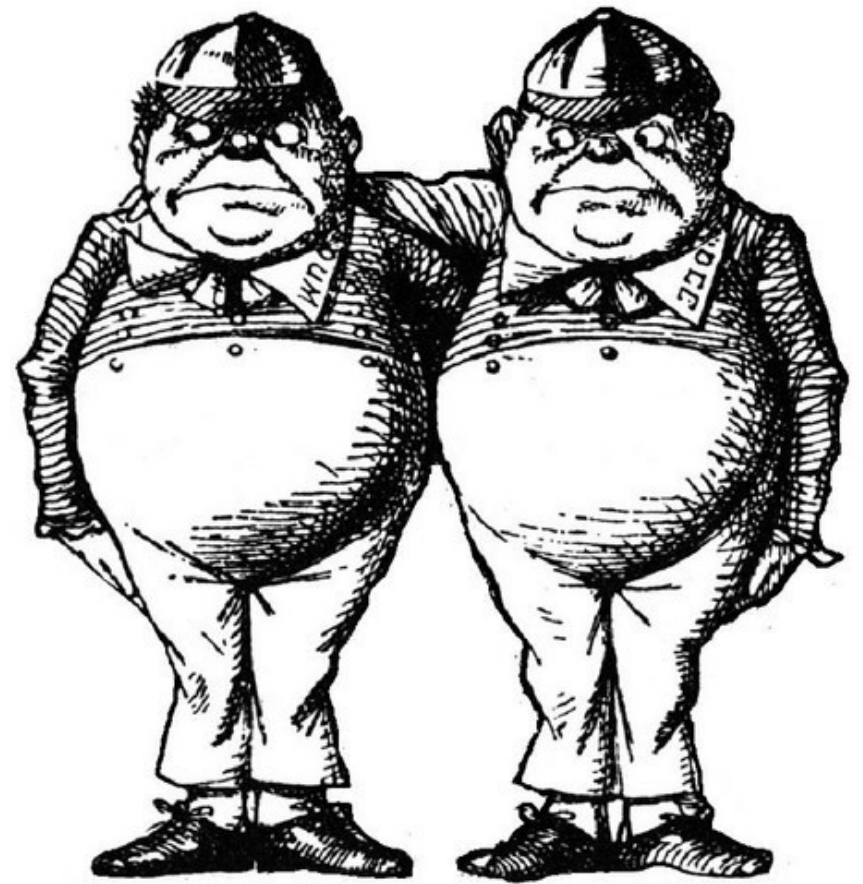
## The Making Of

Cédric Brun <[cedric.brun@obeo.fr](mailto:cedric.brun@obeo.fr)> / @bruncedric





# Personnas



# The Life of Alice

- Prototyping w. Customer
- Focused Diagrams
- Doc and Notes
- Reviewing Design



# Demo Wrap-up

Semi synchronized class diagram

Layers : documentation, validation

Import elements in the diagram one by one

Slightly redesigned graphical style

A lot of shortcuts, smarter palette, smarter edit

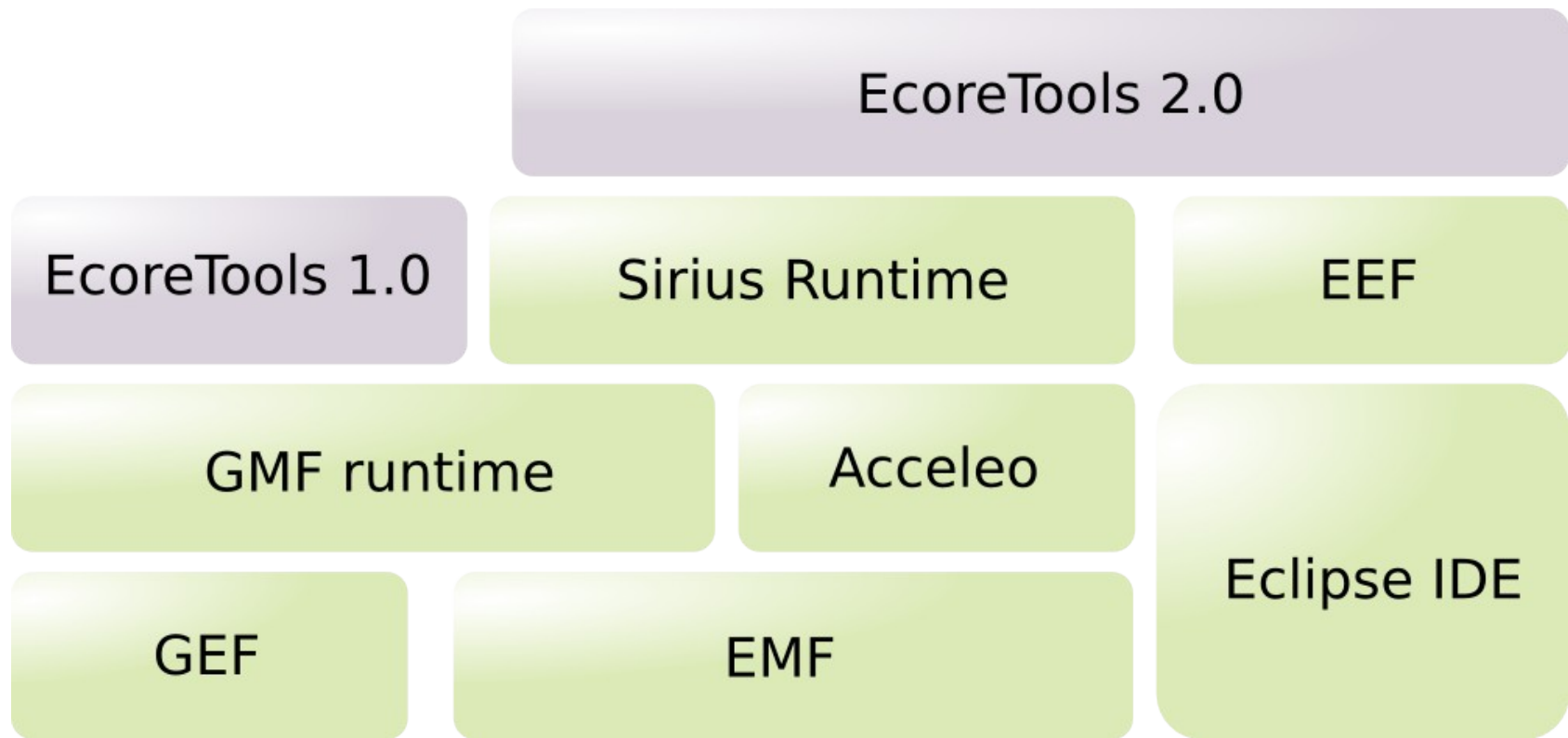
A documentation Table

And many more...



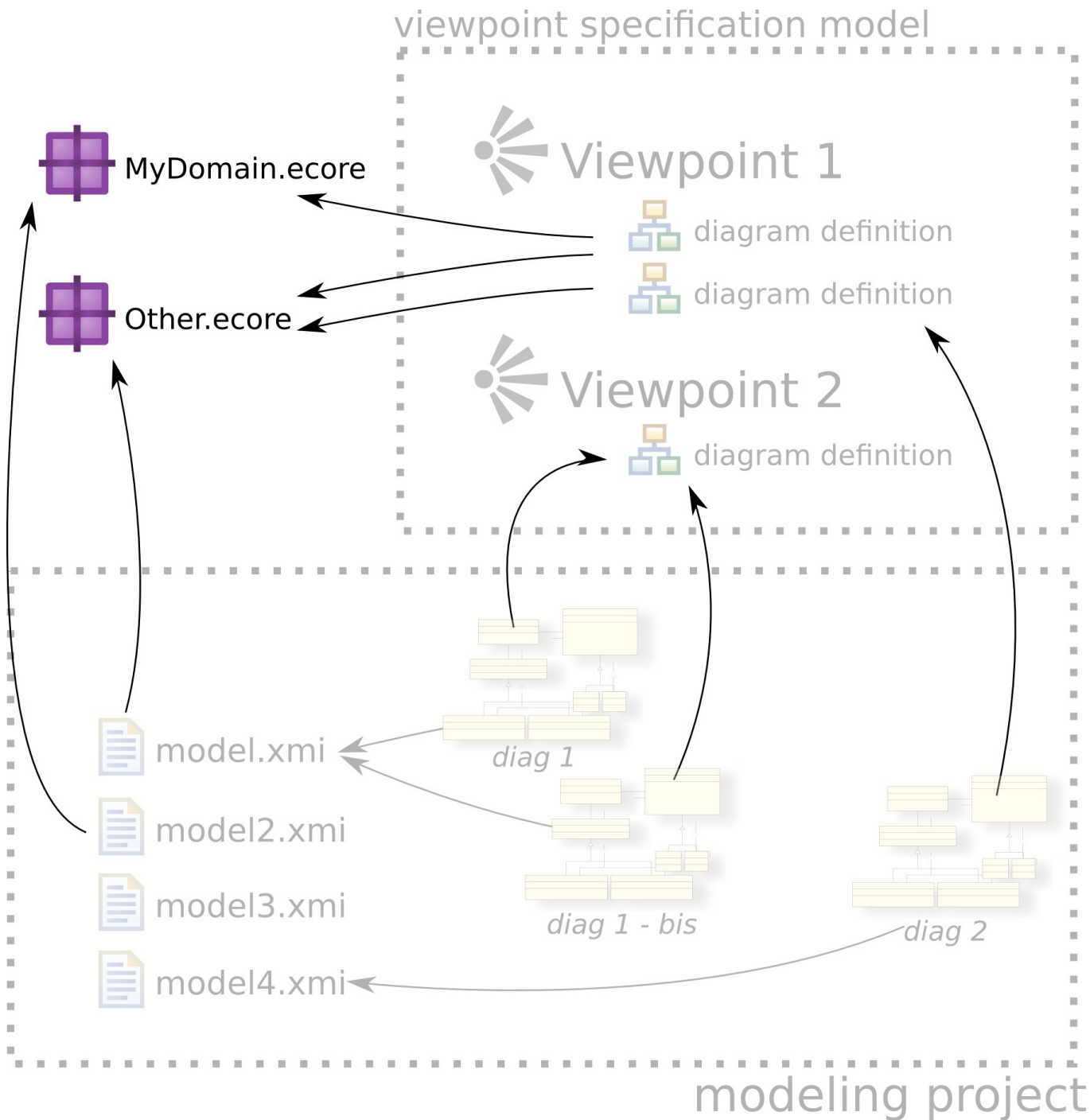


**The Making Of**

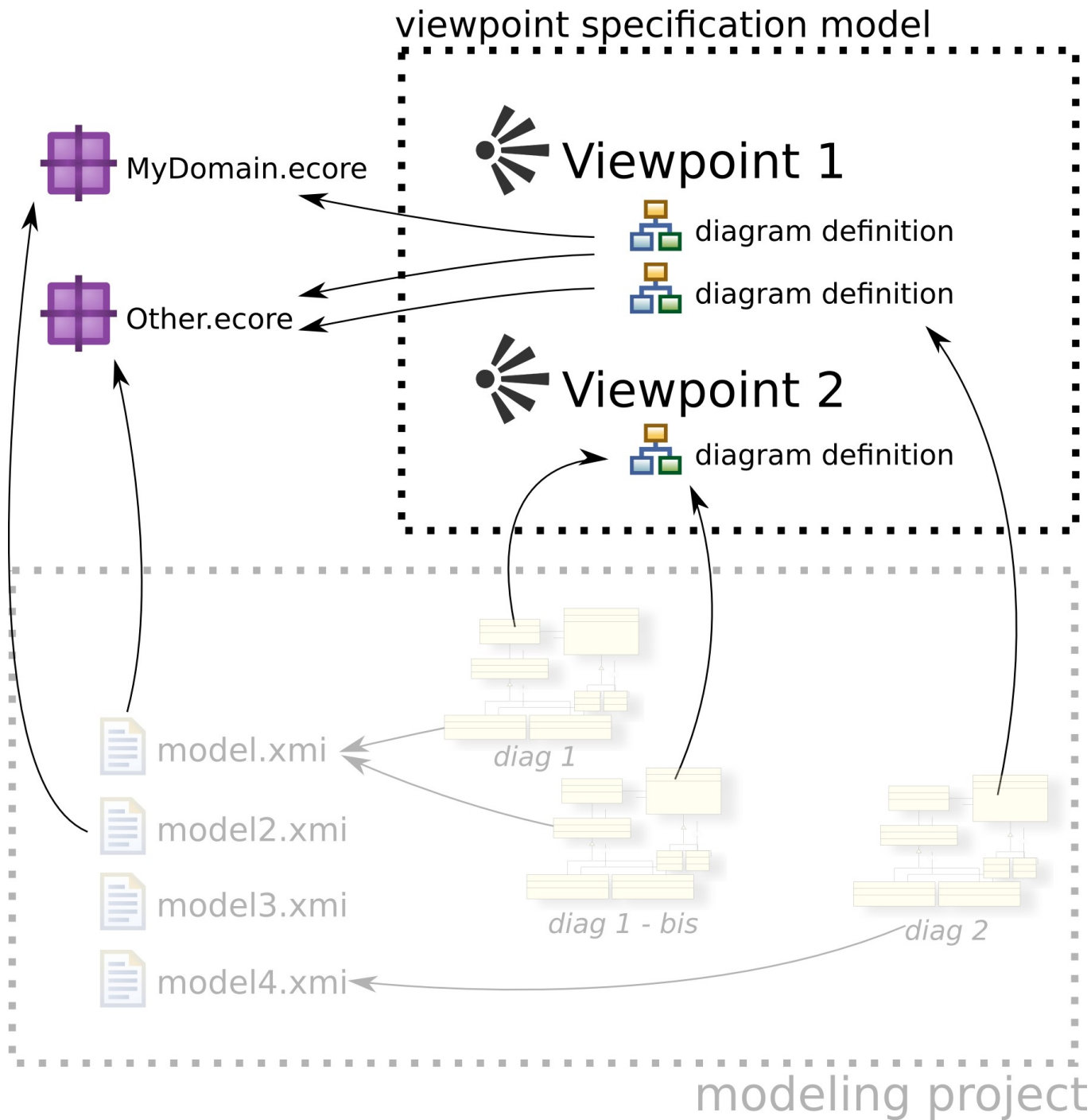




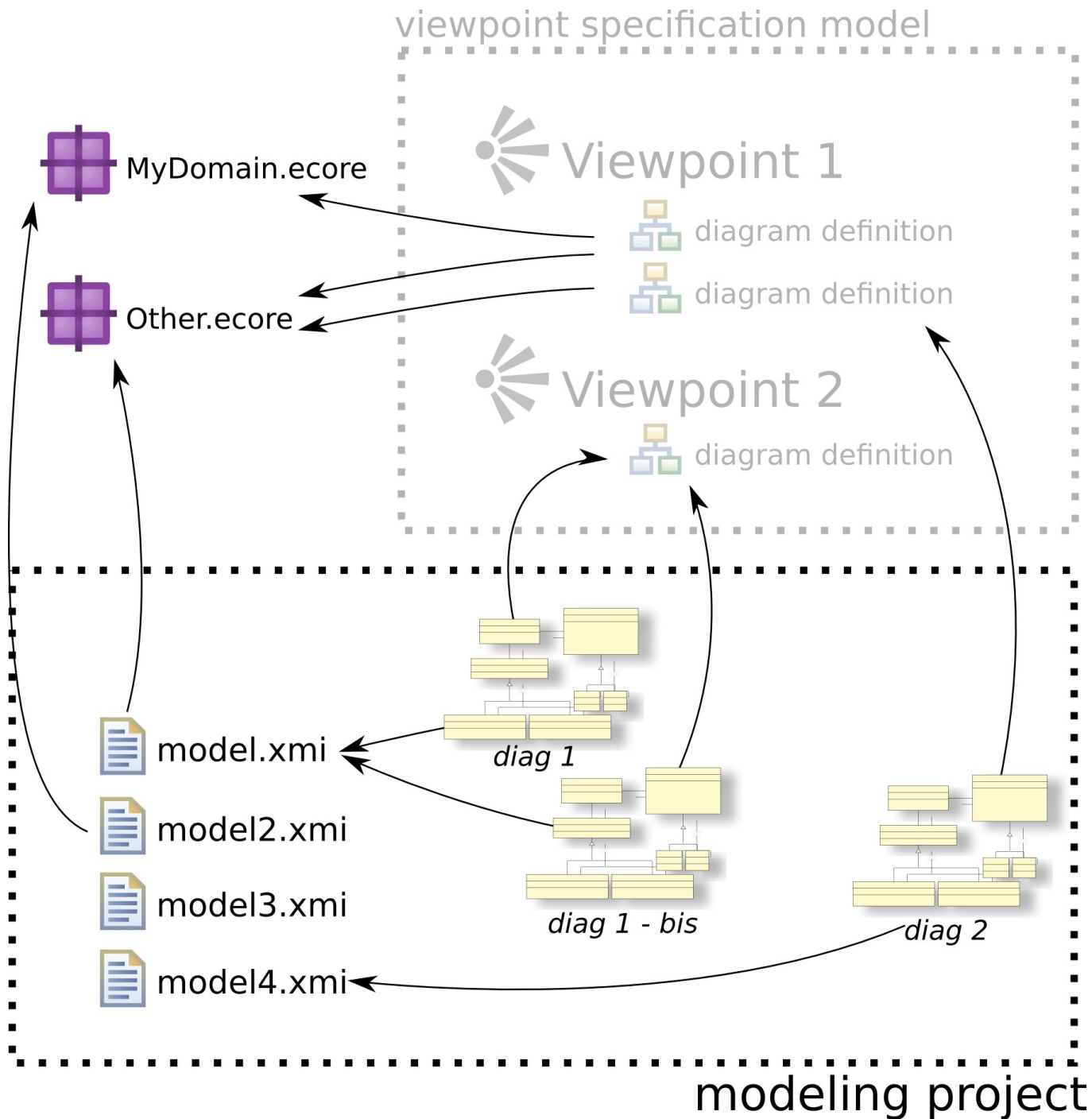
# a dedicated Tooling ?



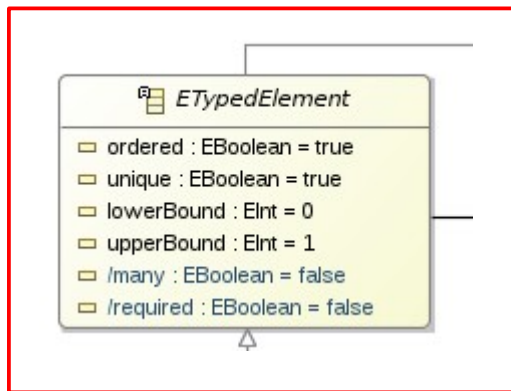
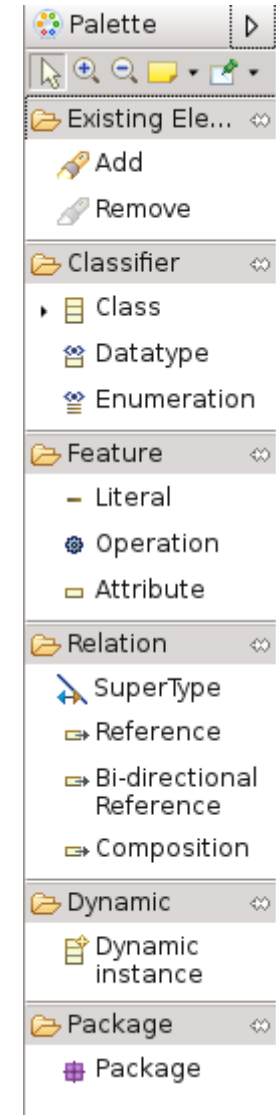
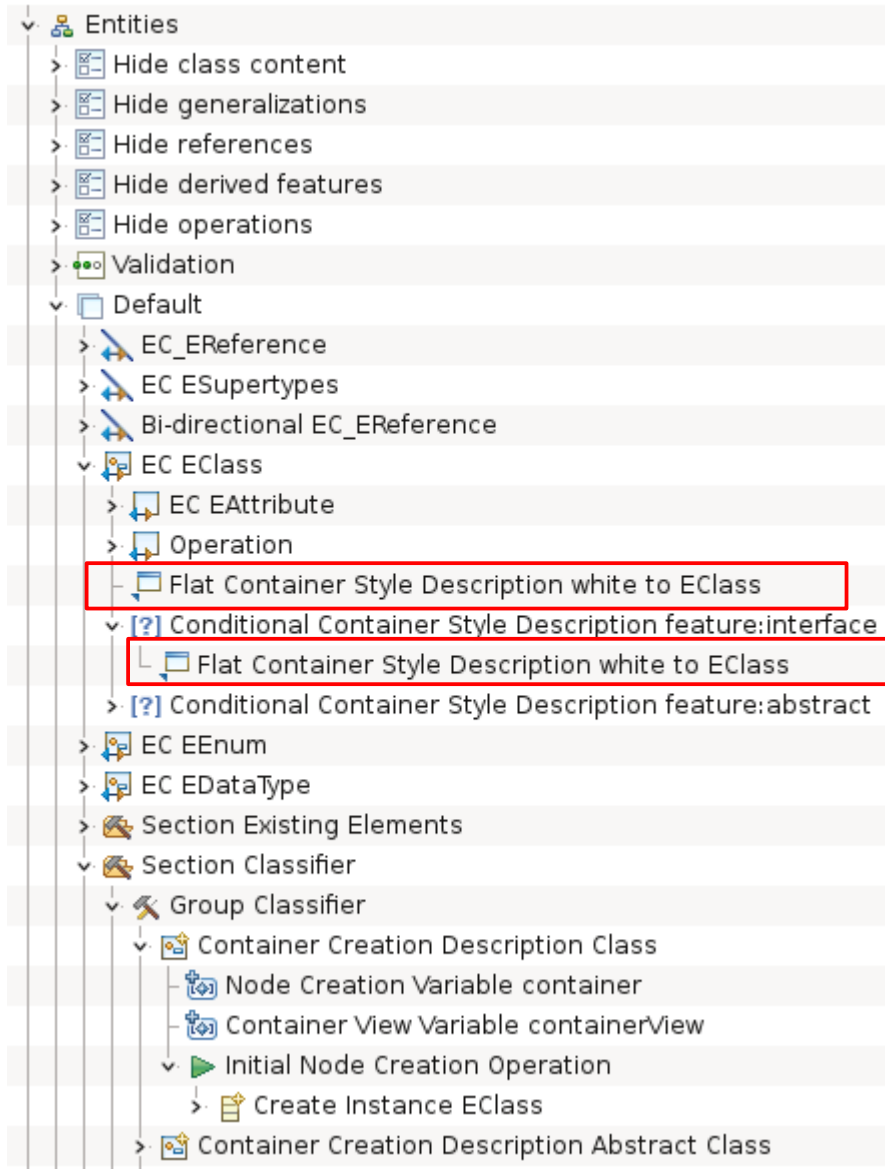
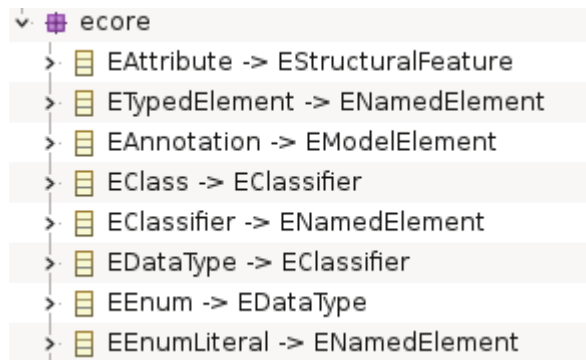
# a dedicated Tooling ?



# a dedicated Tooling ?

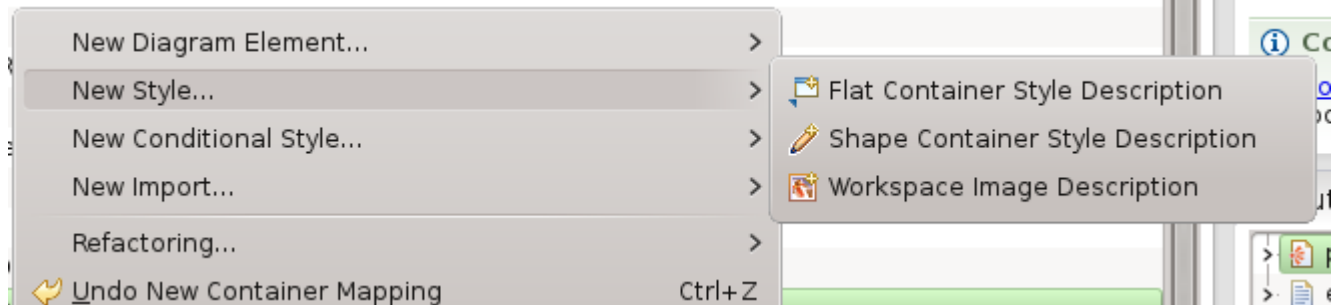
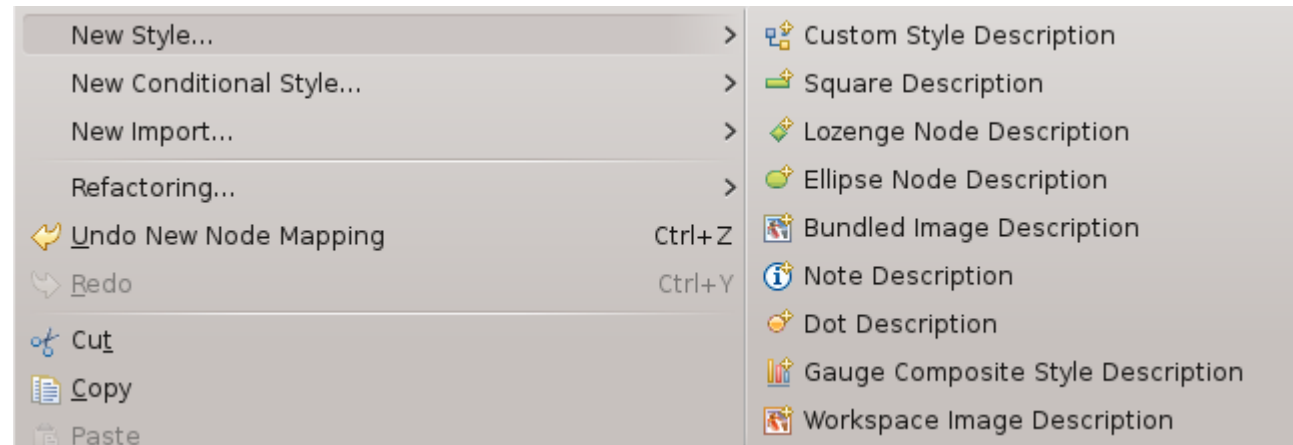


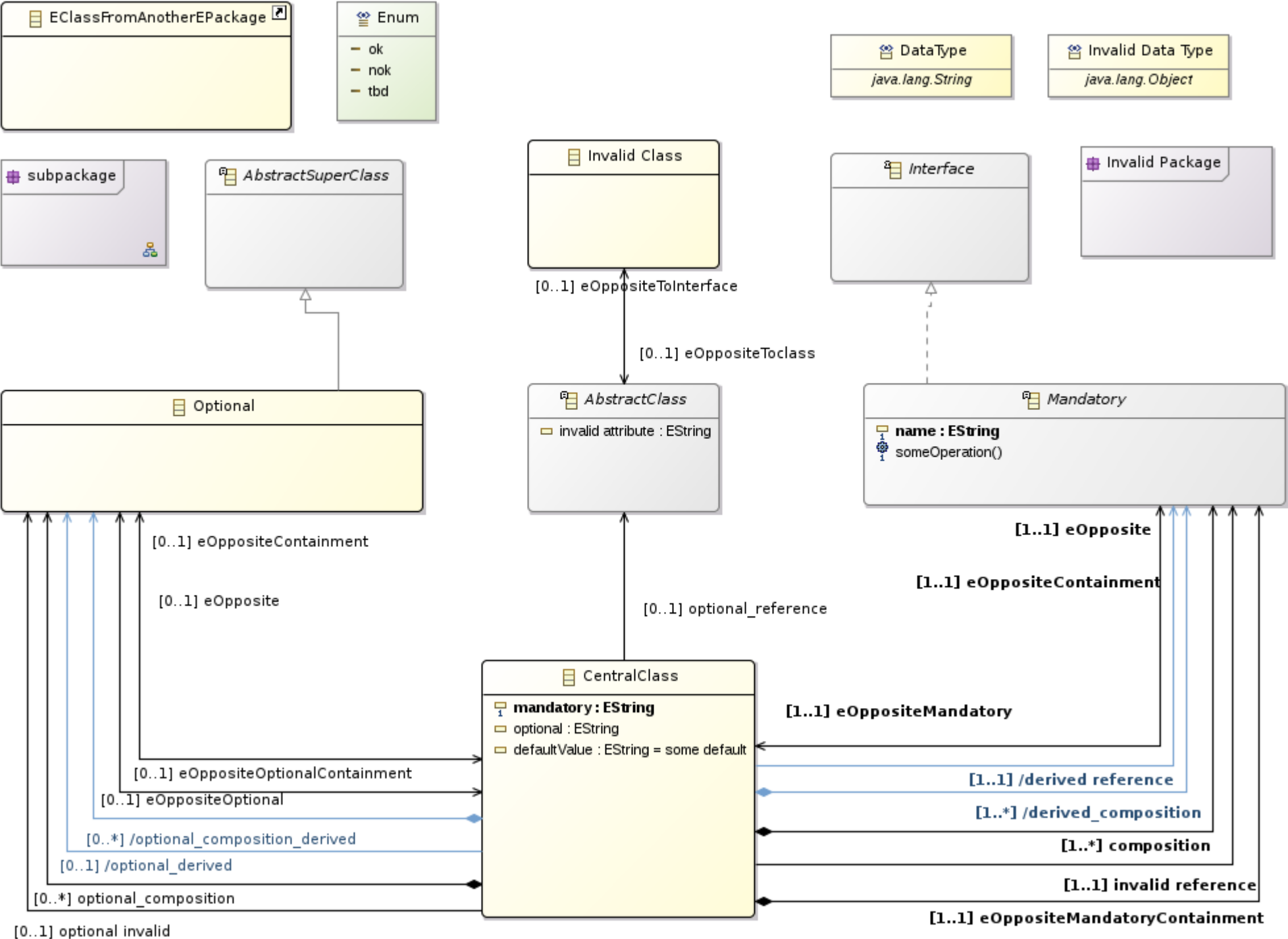
# Diagram definition - styles



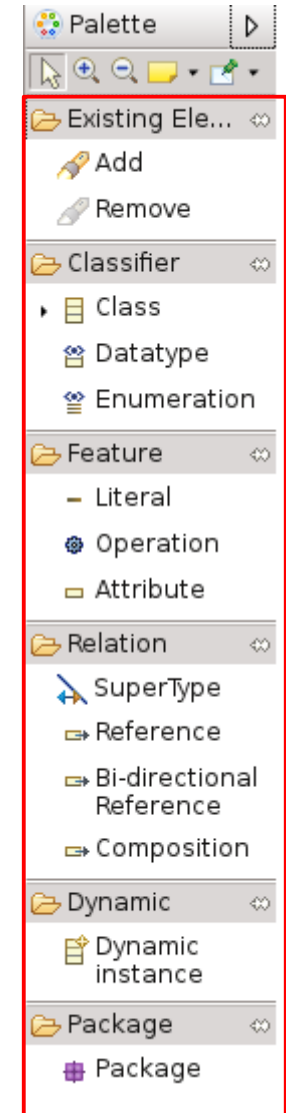
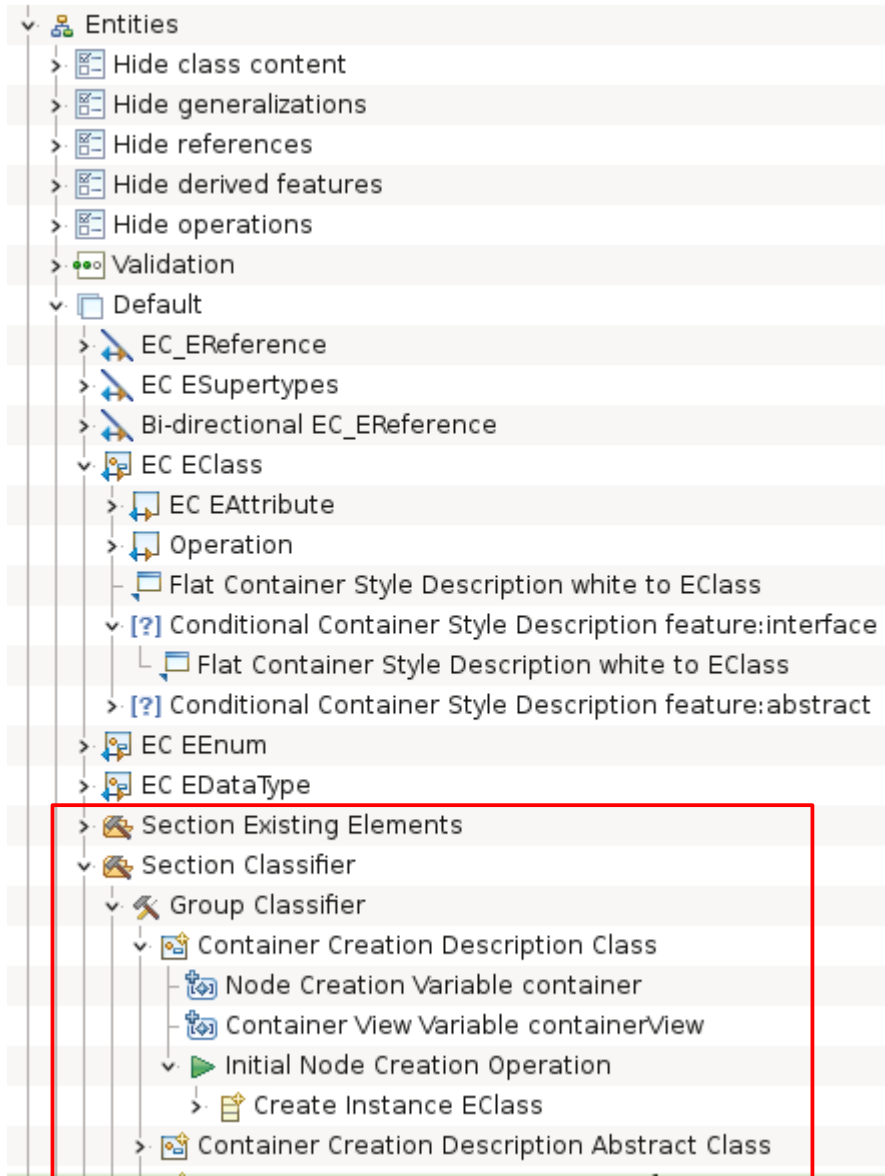
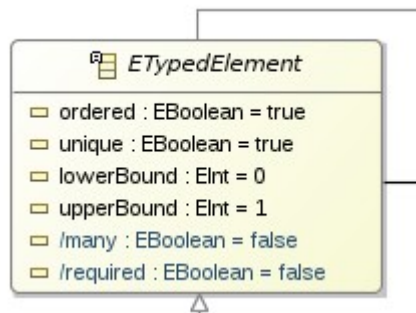
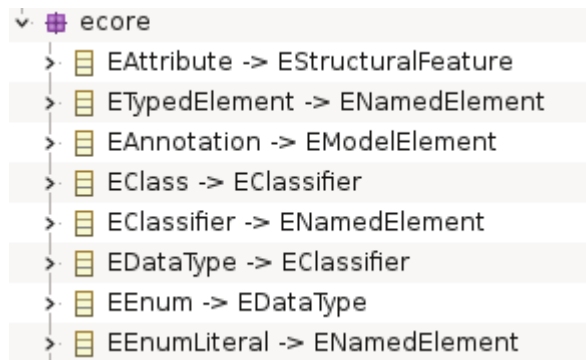


# Styles

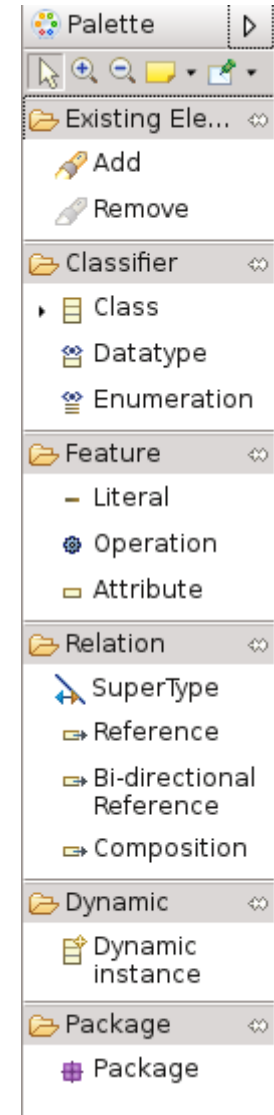
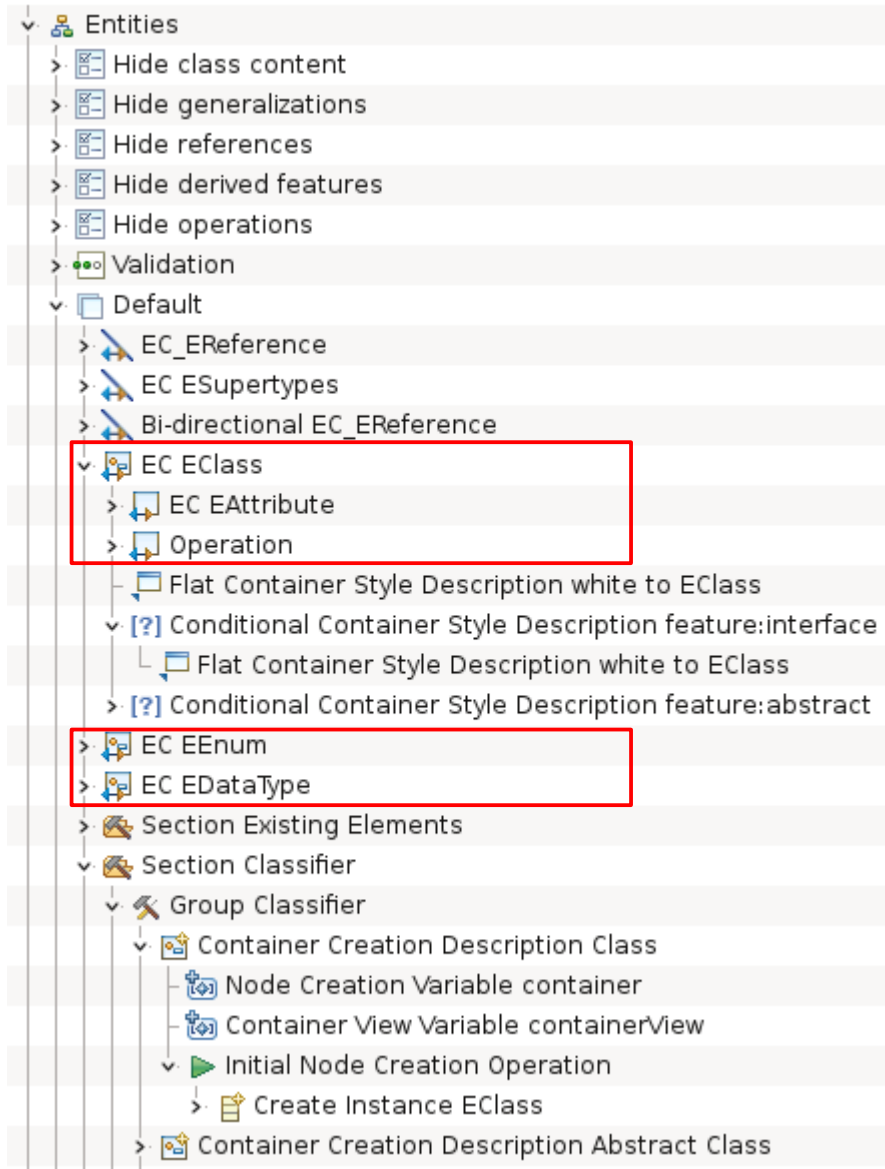
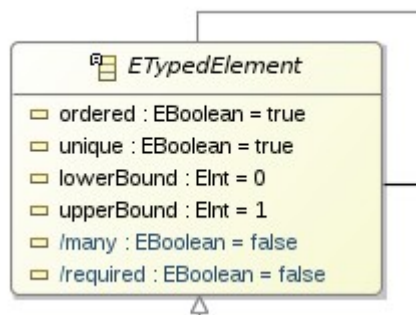
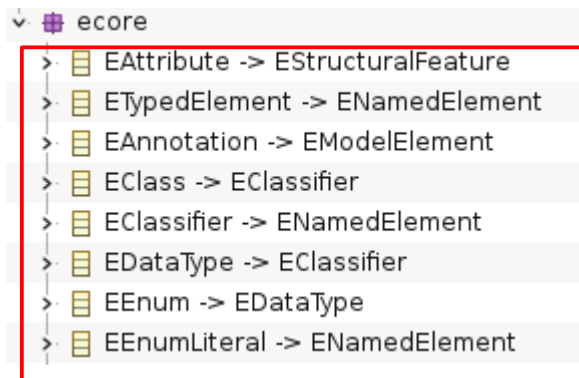




# Diagram definition - tools

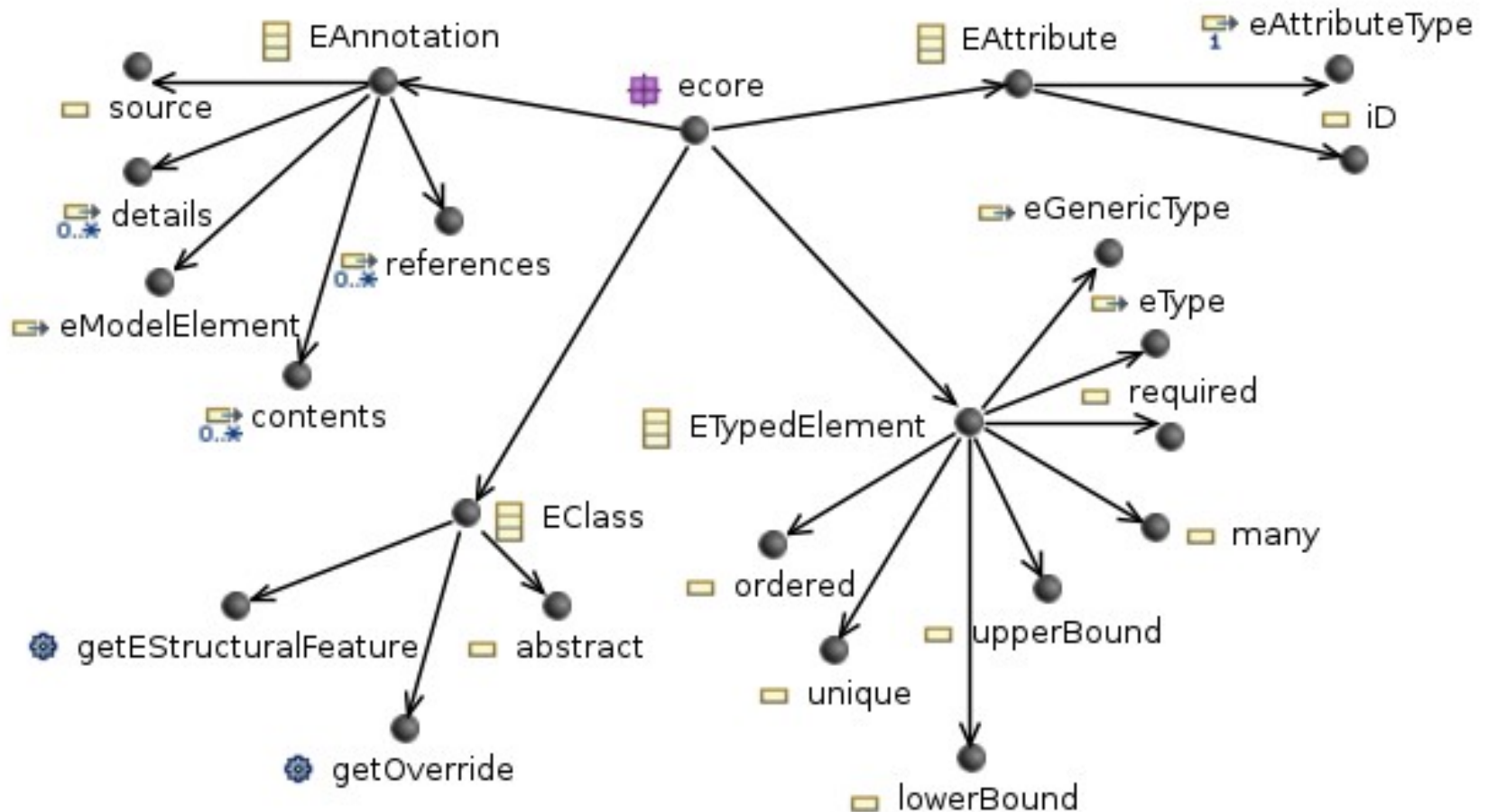


# Diagram definition - mappings

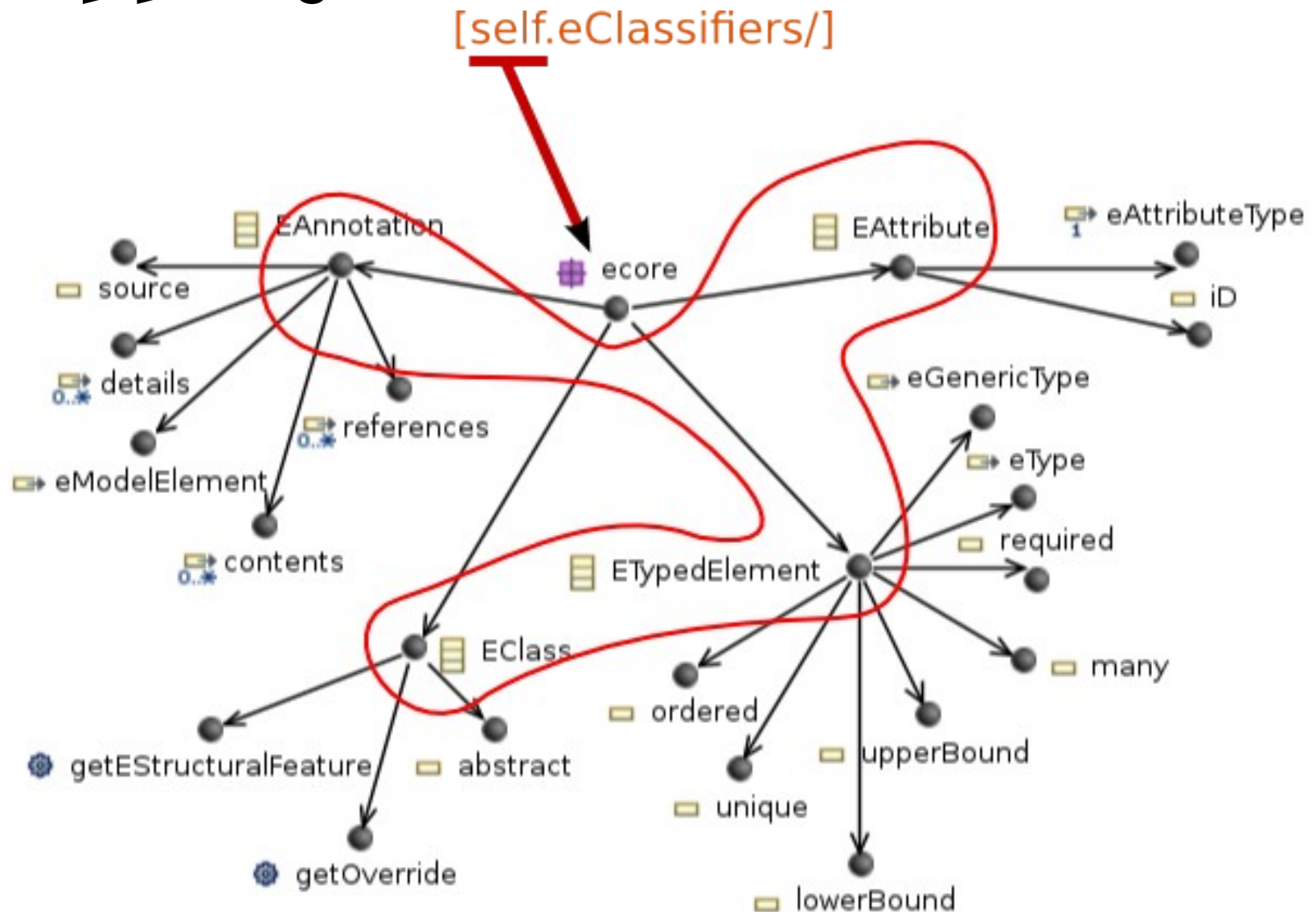




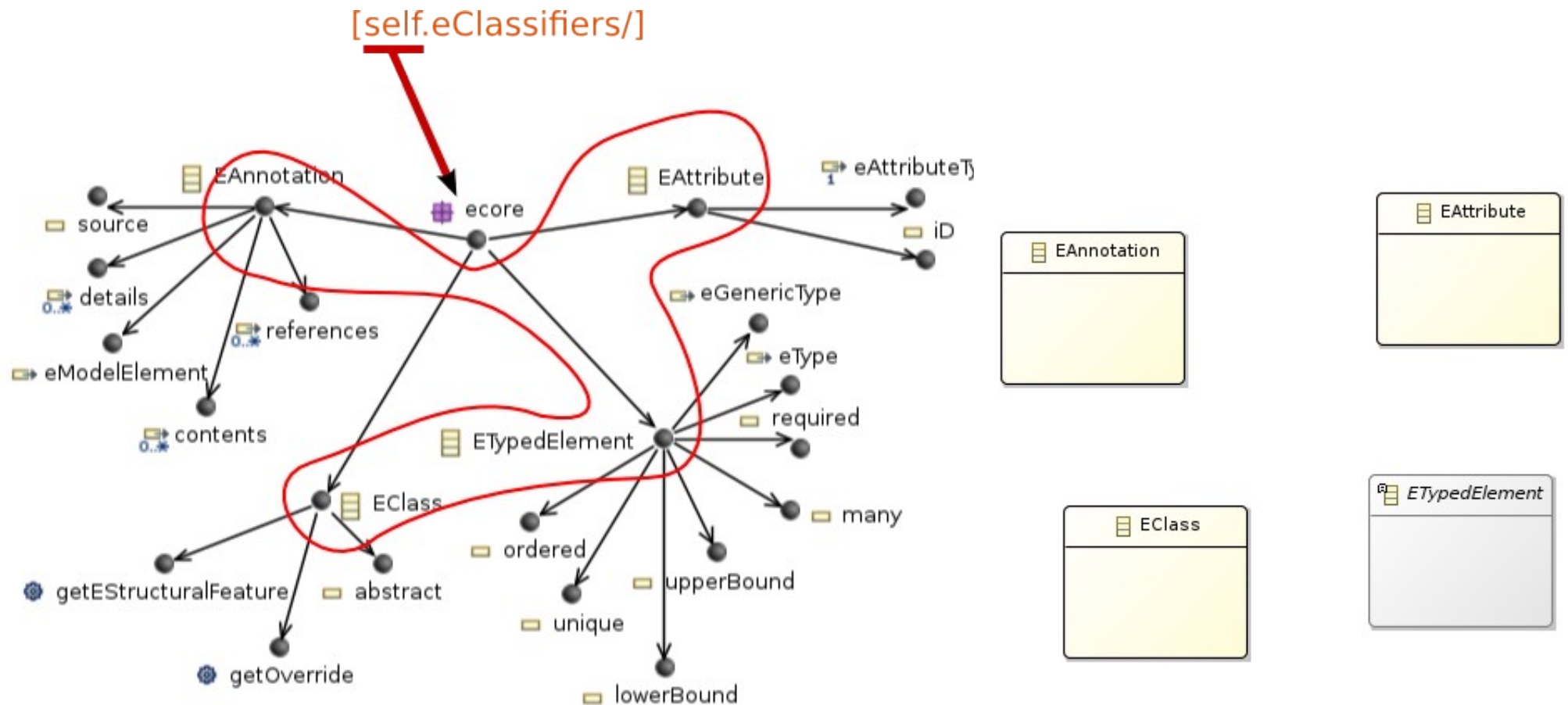
# A Mapping



# A Mapping



# A Mapping




# Synchronization Policy





# Mapping for EClasses

 EC EClass

**General**  
Import  
Documentation  
Behavior  
Advanced


**Id\*:**  ? **Label:**  ?

**Domain Class\*:**  ?

**Children Presentation\*:** ☐ FreeForm ☒ List ☐ Horizontal Stack ☐ Vertical Stack

Semantic Candidates Expression:  ?

Associated Elements Expression:  ?


 EC EClass

**General**  
Import  
Documentation  
Behavior  
**Advanced**


**Synchronization:** ☒ Not synchronized ☐ Unsynchronizable ☐ Synchronized ?

**Precondition Expression:**  ?

# Mapping for EReferences

 EC\_EReference

General	Id*: <input type="text" value="EC_EReference"/>	Label: <input type="text" value="EC_EReference"/>
Path	Source Mapping*: <input type="text" value="EC EClass"/>	
Documentation	Domain Class*: <input type="text" value="EReference"/>	
Behavior	Source Finder Expression: <input type="text" value="feature:eContainer"/>	
Advanced	Target Mapping*: <input type="text" value="EC EClass"/>	
	Target Finder Expression*: <input type="text" value="feature:eType"/>	
	Semantic Candidates Expression: <input type="text" value="[diagram.getDisplayedEClasses().oclAsType(ecore::EClass).eAllReferences-&gt;flatten()]/"/>	
	Associated Elements Expression: <input type="text" value="var:self"/>	

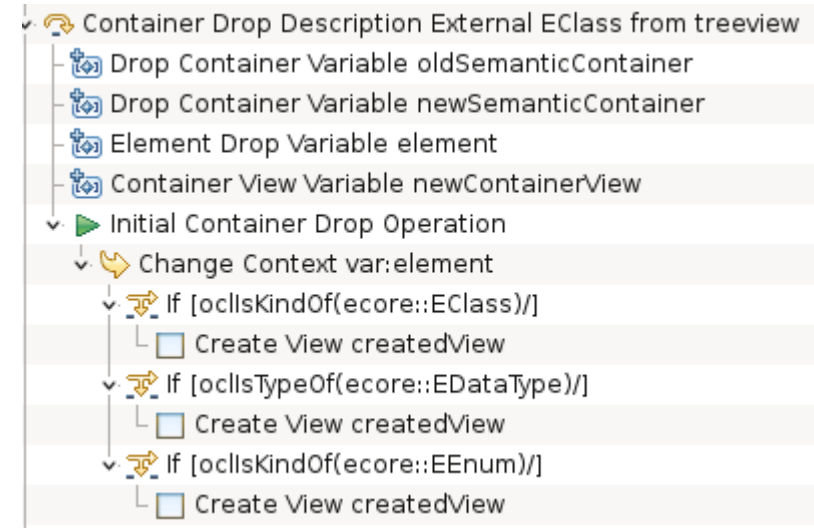
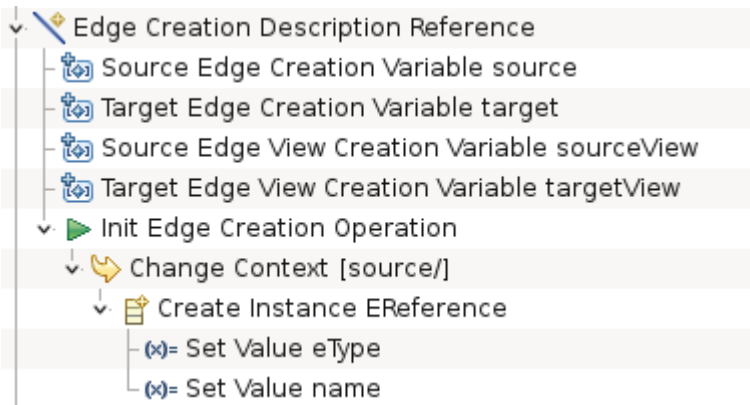
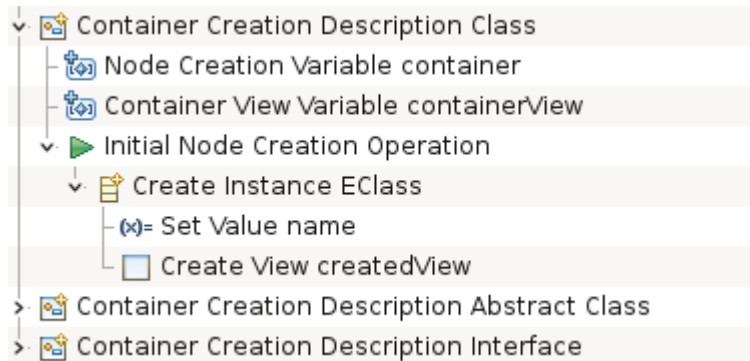
 EC\_EReference

General	Synchronization: <input type="radio"/> Not synchronized <input type="radio"/> Unsynchronizable <input checked="" type="radio"/> Synchronized
Path	Target Expression: <input type="text" value=""/>
Documentation	
Behavior	
Advanced	Precondition Expression: <input type="text" value="service:noEOpposite"/>

# Break 1

Tooling is specified in a model and interpreted @ runtime  
The coupling to the semantic models is kept low thanks to queries  
Synchronization Policy can be specified on a per mapping basis.  
Queries can be OCL/Acceleo, or even simple Java calls

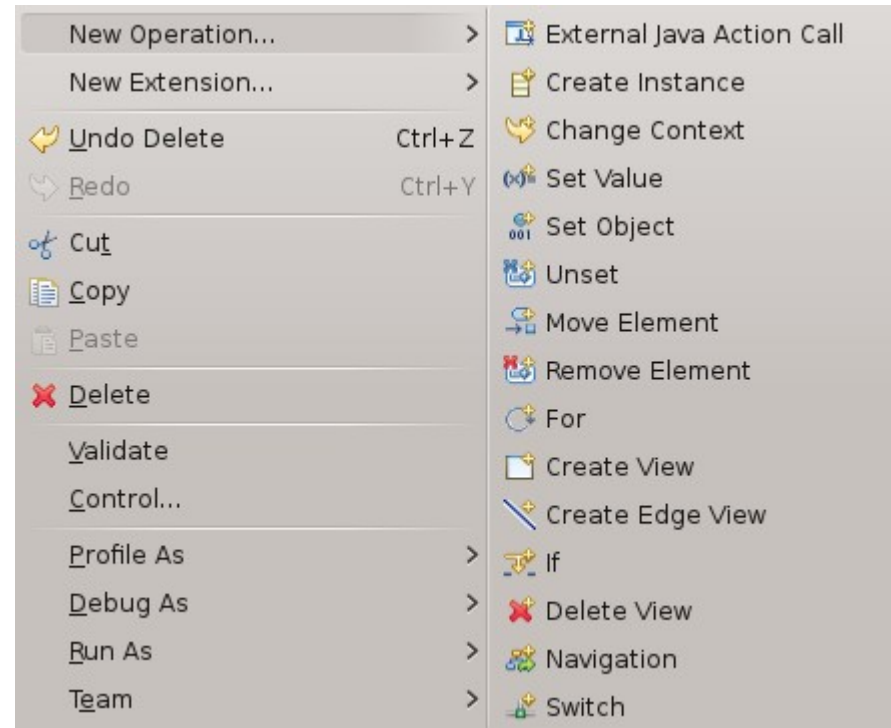
# the Tools





# Action language

Create  
Reconnect  
drag&drop  
Direct edit  
Double click  
Delete  
Popup actions  
Validation quickfixes



## Set Value name

General	Feature Name*:	name
	Value Expression:	['NewEClass' + eContainer().eContents(ecore::EClass)->size()/]

## Create Instance EClass

General	Reference Name*:	eClassifiers
	Type Name:	EClass
	Variable Name:	instance

# Label edition

« Something » => change name of feature

«:SomeType » => only change the eType

« 1 » => only set cardinality to 1..x

« \* » => only set cardinality to x..\*

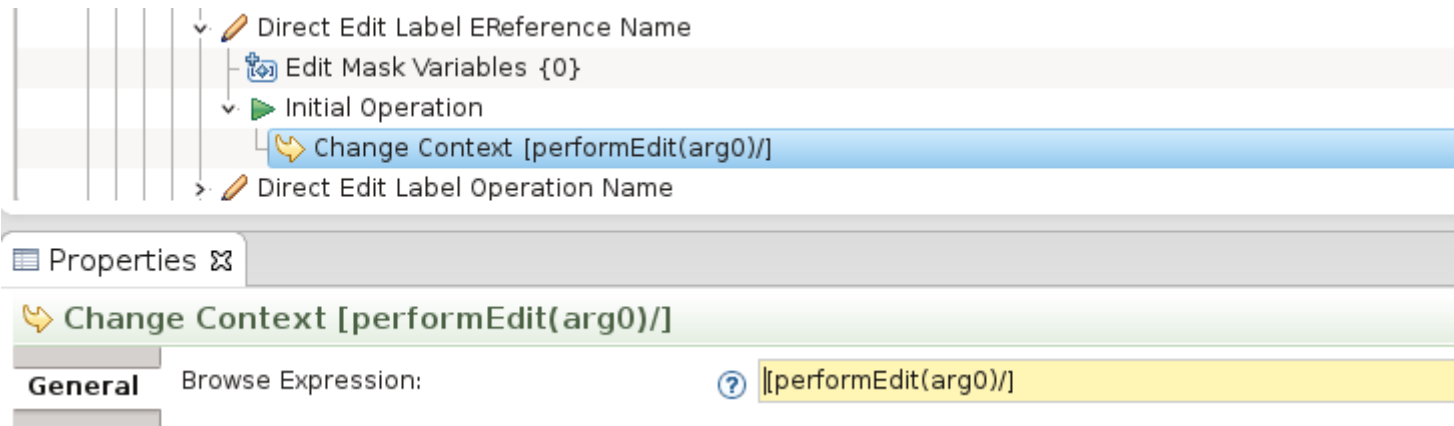
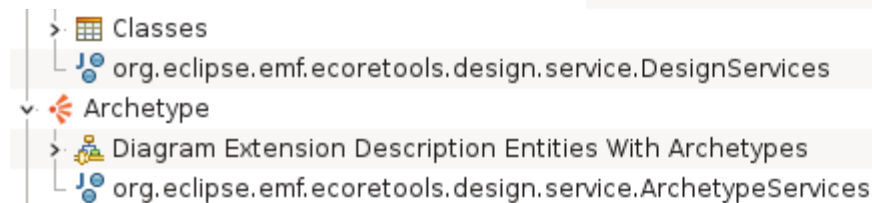
« /Something » => make the feature derived

« = something » => set the default value literal

[...]

# Label edition (bis)

```
public EReference performEdit(EReference ref, String editString) {  
    if ("0".equals(editString.trim())) {  
        ref.setLowerBound(0);  
    } else if ("1".equals(editString.trim())) {  
        ref.setLowerBound(1);  
    } else if (CARDINALITY_UNBOUNDED.equals(editString.trim())) {  
        ref.setUpperBound(-1);  
    } else if (CARDINALITY_UNBOUNDED_ALTERNATIVE.equals(editString.trim())) {  
        ref.setUpperBound(-1);  
    } else {  
        editName(ref, editString);  
        editCardinality(ref, editString);  
    }  
    return ref;  
}
```

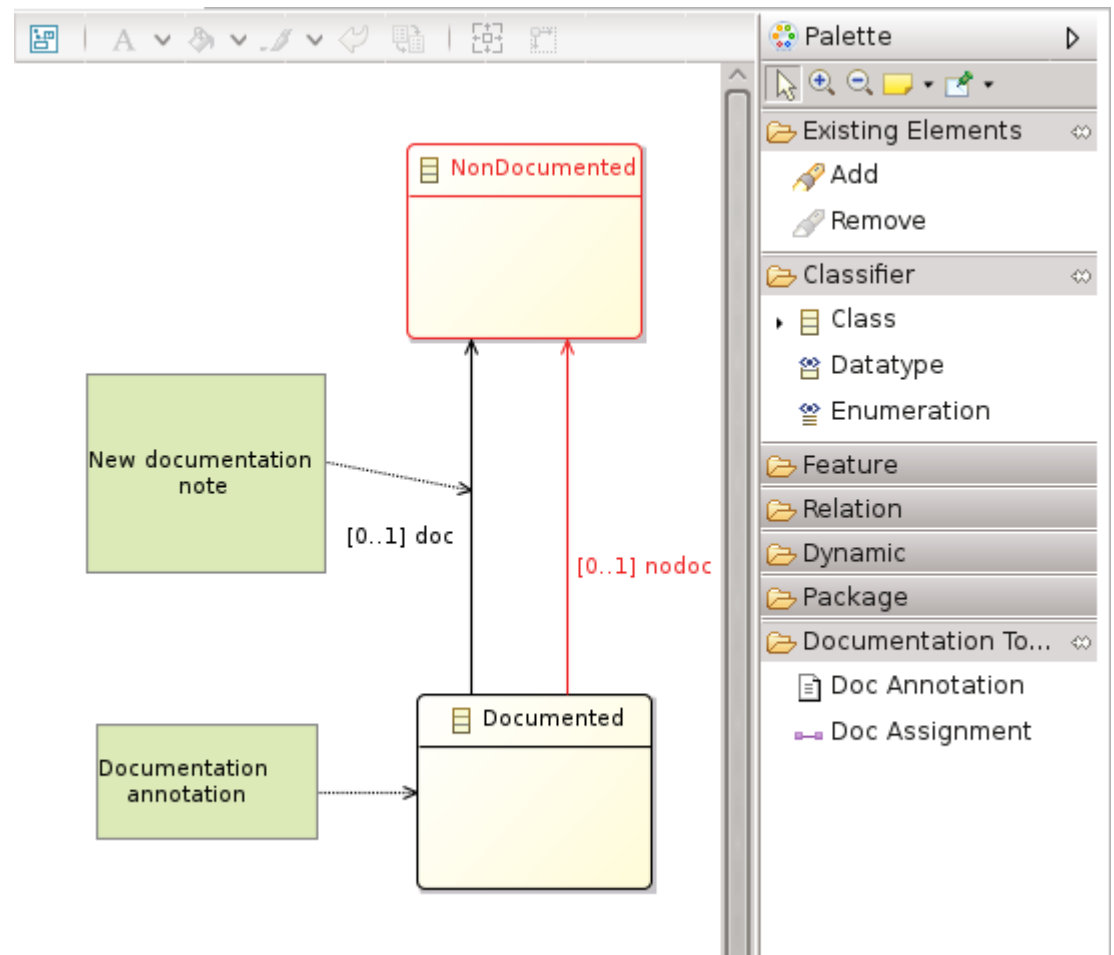
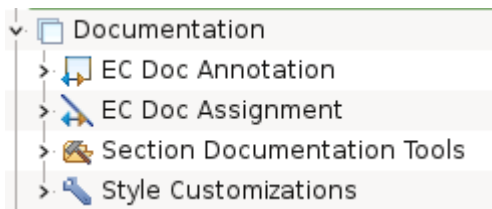


# Documentation Layer

Add new Mappings

Add new Tools

Customize Styles



# Break 2

Because of the low coupling with semantic model, one has to define the effects of the interactions

A Simple Action language exists for this  
(create/update/delete/ of elements and views)

Calling Java is definitely an option

Expressiveness is pretty rich. Diagram have Layers, filters, tools, decorators



# The Life of Cédric

- Creating EcoreTools
- Consistency
- Performance
- Building
- Testing
- Upgrading



# Checking Consistency

```
@RunWith(value = Parameterized.class)
public class EdgeMappingConsistencyTests {

    private EdgeMapping underTest;

    public EdgeMappingConsistencyTests(EdgeMapping edgemapping) {
        this.underTest = edgemapping;
    }

    @Test
    public void hasReconnect() {
        assertTrue("Edge Mapping +" + underTest.getName()
            + "has no reconnect tool",
            underTest.getReconnections().size() > 0);
    }
}
```

The screenshot shows the Eclipse IDE interface with the JUnit test runner. The Package Explorer on the left shows the test class `org.eclipse.emf.ecoretools.design.tests.EdgeMappingConsistencyTests`. The JUnit runner window displays the following information:

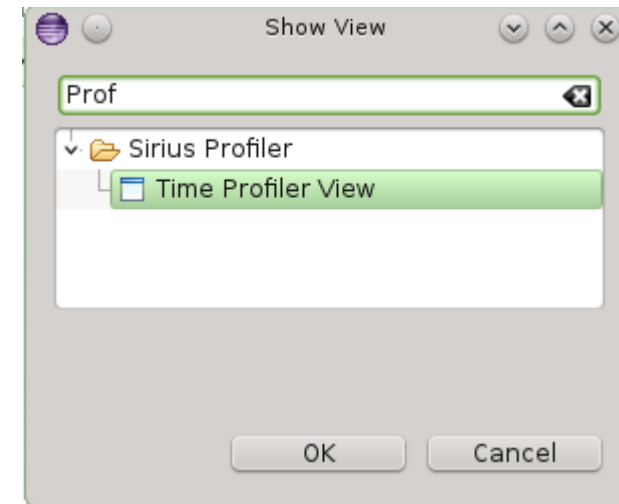
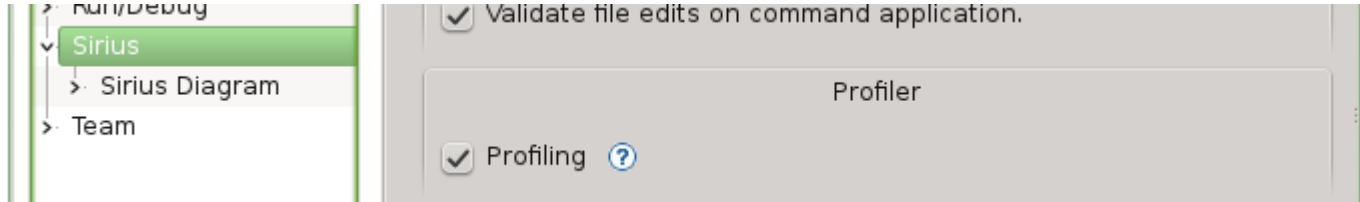
- Finished after 0,03 seconds
- Runs: 5/5
- Errors: 0
- Failures: 2

The test results tree shows five test cases, each with a sub-test `hasReconnect`. The first four tests passed, but the fifth test case, `[4] (0,003 s)`, failed. The failure trace for the failed test is as follows:

```
java.lang.AssertionError: Edge Mapping +EC Doc Assignmenthas no reconr
at org.eclipse.emf.ecoretools.design.tests.EdgeMappingConsistencyTests.
at org.eclipse.pde.internal.junit.runtime.RemotePluginTestRunner.main(Ren
at org.eclipse.pde.internal.junit.runtime.PlatformUITestHarness$1.run(Platt
at java.lang.Thread.run(Unknown Source)
```

Your own checkstyle for Viewpoint Specification model

# Measuring performances



Measure first  
Think about your queries  
Use the cross referencer

The screenshot shows the 'Time Profiler View' window. It contains a table with the following columns: Task Category, Task Name, Time (ms), Time (hh:mm:ss,ms), Occurrences, and Minimum. The table lists various tasks and their performance metrics.

Task Category	Task Name	Time (ms)	Time (hh:mm:ss,ms)	Occurrences	Minimum
Acceleo	feature:eOperations	0	0:0:0,0	36	0
Acceleo	service:getVisibleAnnotations(diagram)	2	0:0:0,2	12	0
Other	Other	22	0:0:0,22	0	1
DDiagram	Get edge's candidates	18	0:0:0,18	24	1
DDiagram	Compute edge source/target views	5	0:0:0,5	48	0
DDiagram	Get edge's candidates	13	0:0:0,13	24	1
Acceleo	feature:eType	0	0:0:0,0	12	0
Acceleo	service:eContainerEContainer	0	0:0:0,0	12	0
Acceleo	feature:eContainer	0	0:0:0,0	12	0
Acceleo	feature:eSuperTypes	0	0:0:0,0	18	0
Acceleo	Check precondition expressions	1	0:0:0,1	24	0
DDiagram	Get node's candidates	6	0:0:0,6	12	0
Other	Other	6	0:0:0,6	0	1
DDiagram	Updating all edges	37	0:0:0,37	24	0
Other	Other	141	0:0:0,141	0	20

# Just an Eclipse Plugin

Language	files	blank	comment	code
Java	18	353	481	1993
XML	5	9	22	194
Maven	1	8	4	31
HTML	2	5	0	25
SUM:	26	375	507	2243

Tycho build  
JUnit and SWTBot Tests

# Break 3

Model @ runtime means some things are easier.  
You have to care about performances to have good ones  
Modeling Workbenches are just Eclipse Plugins.



# Roadmap

Luna M3 : built and aggregated

Luna M4 : installable and usable in general

Luna M5 : better property views, other stuff

**Luna : Sirius based on GMF based modelers  
shipped with Luna**

# **The Sirius BOF**

## **The Booth**

