THALES

EGF Tutorial
Pattern Trace

Benoît Langlois – Thales/TGS

EGF - Thales Global Services

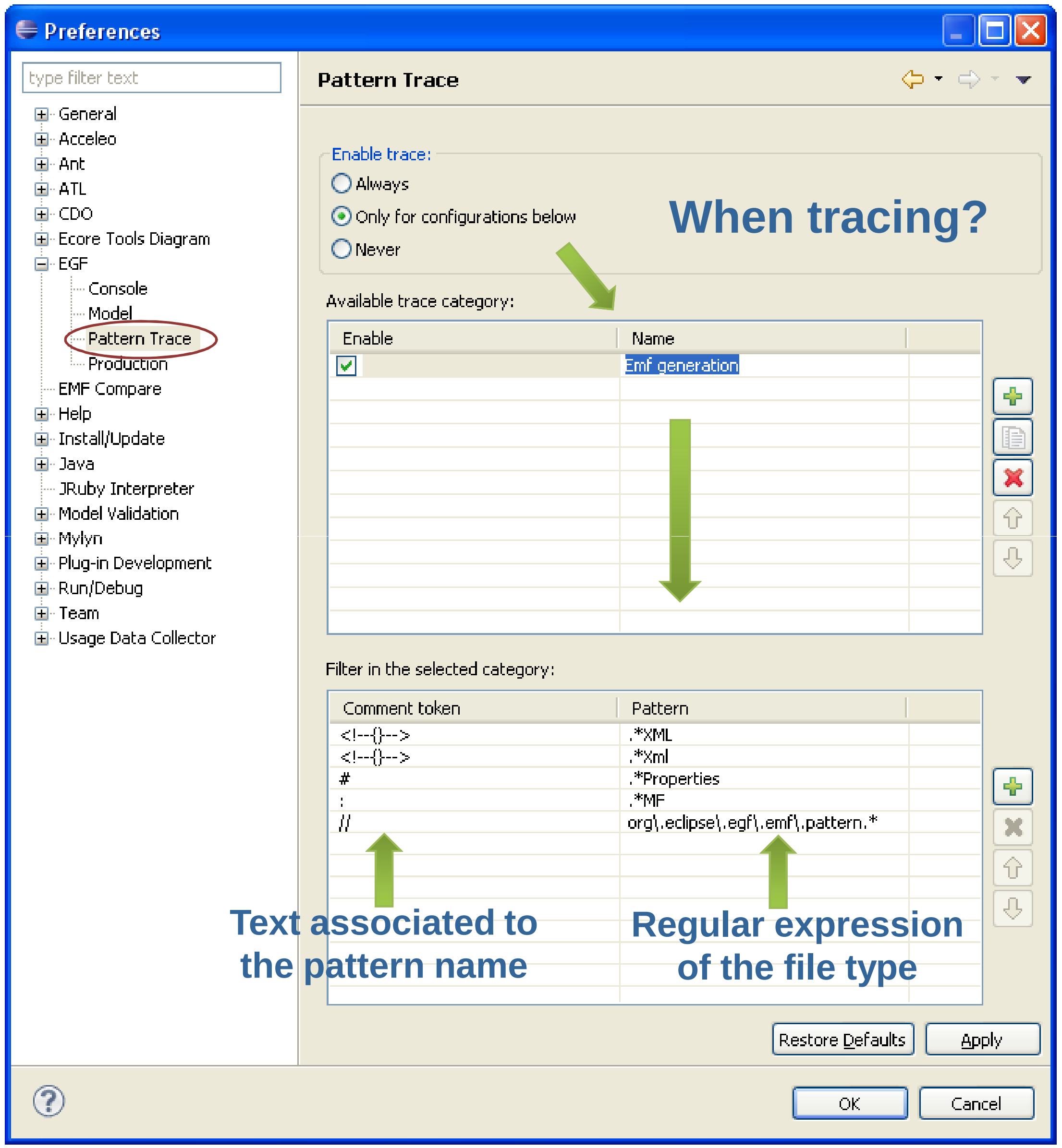
The issue

- **Identification of involved templates for M2T (Model-to-Text) transformations:**
  - For large textual generations, the result of a M2T transformation cannot identify the set of templates selected and involved during a transformation
  - This becomes more difficult when a transformation involves inheritance, delegation, or when the language is declarative

- **For this reason, a trace mechanism was introduced in EGF in order to identify and track the set of patterns (with their templates) involved during a M2T transformation**

THALES

Setting the EGF Preferences



Pattern Trace

Enable trace:
- ○ Always
- ● Only for configurations below
- ○ Never

When tracing?

Available trace category:

Enable	Name
✓	Emf generation

Filter in the selected category:

Comment token	Pattern
<!--{}-->	.*XML
<!--{}-->	.*Xml
#	.*Properties
:	.*MF
//	org\.eclipse\.egf\.emf\.pattern.*

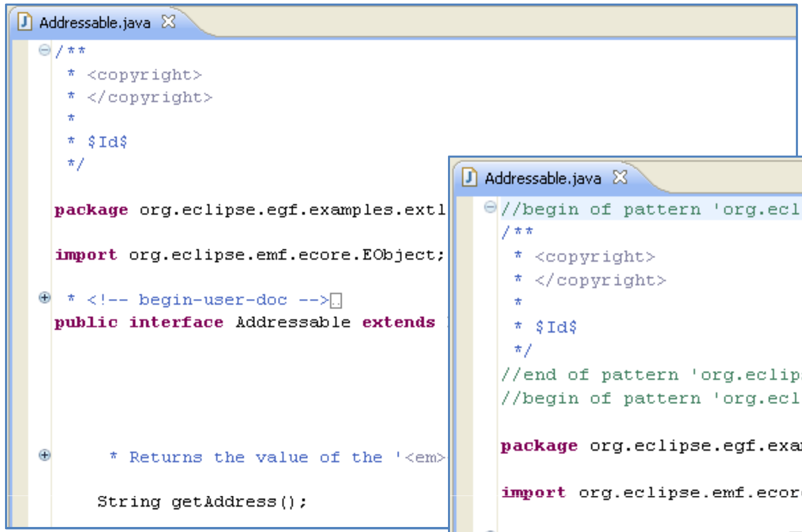
Text associated to the pattern name

Regular expression of the file type

THALES

Example – EMF Generation - Class

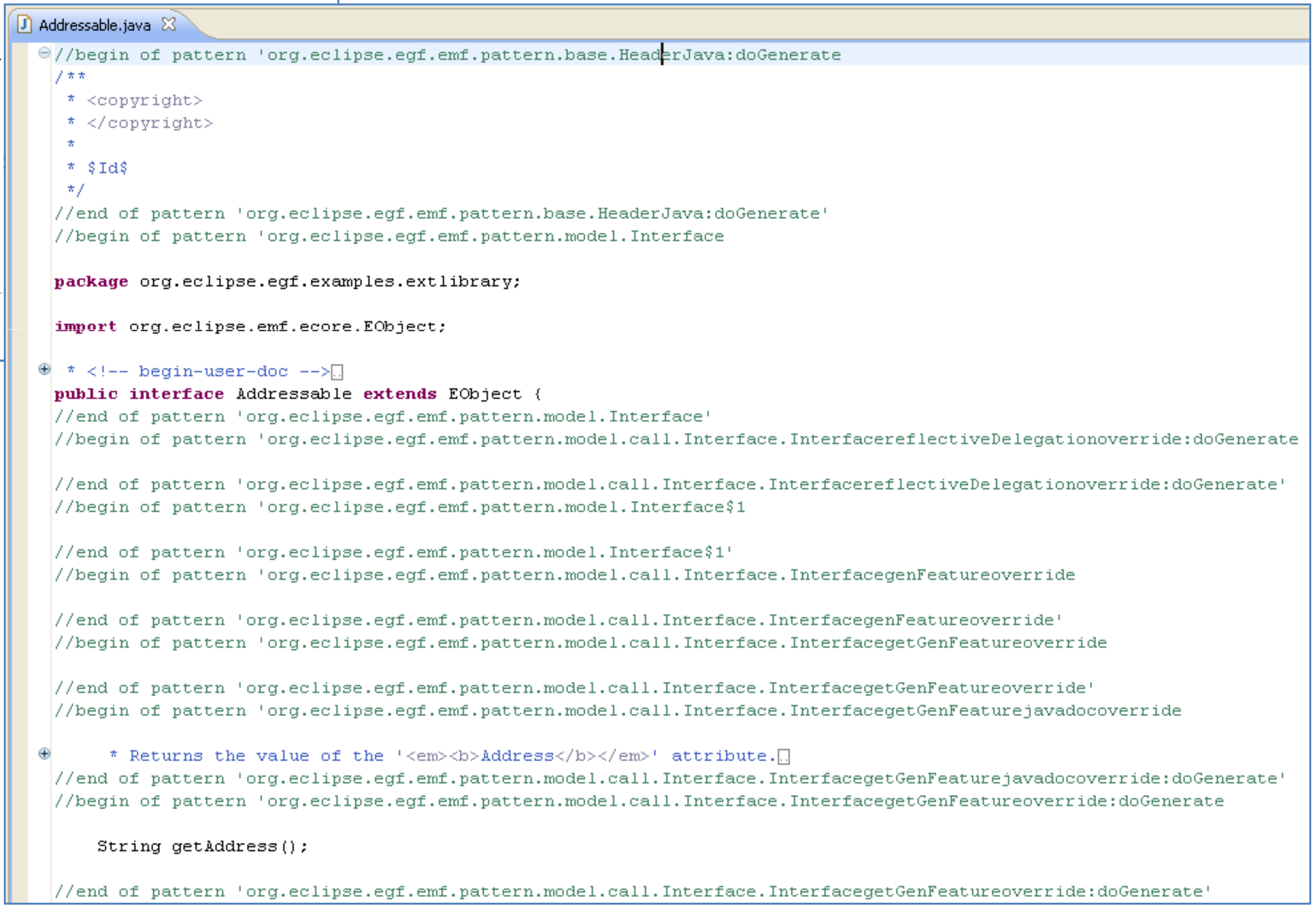
Without Trace



Addressable.java

```
/**
 * <copyright>
 * </copyright>
 *
 * $Id$
 */

package org.eclipse.egf.examples.extl

import org.eclipse.emf.ecore.EObject;

 * <!-- begin-user-doc -->
public interface Addressable extends



    * Returns the value of the '<em>

    String getAddress();
```

With Trace

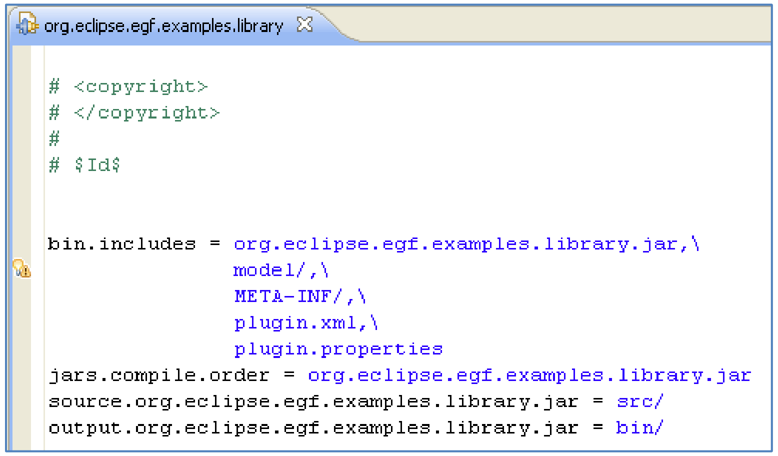


Addressable.java

```
//begin of pattern 'org.eclipse.egf.emf.pattern.base.HeaderJava:doGenerate
/**
 * <copyright>
 * </copyright>
 *
 * $Id$
 */
//end of pattern 'org.eclipse.egf.emf.pattern.base.HeaderJava:doGenerate'
//begin of pattern 'org.eclipse.egf.emf.pattern.model.Interface

package org.eclipse.egf.examples.extlibrary;

import org.eclipse.emf.ecore.EObject;

 * <!-- begin-user-doc -->
public interface Addressable extends EObject {
//end of pattern 'org.eclipse.egf.emf.pattern.model.Interface'
//begin of pattern 'org.eclipse.egf.emf.pattern.model.call.Interface.InterfacereflectiveDelegationoverride:doGenerate

//end of pattern 'org.eclipse.egf.emf.pattern.model.call.Interface.InterfacereflectiveDelegationoverride:doGenerate'
//begin of pattern 'org.eclipse.egf.emf.pattern.model.Interface$1

//end of pattern 'org.eclipse.egf.emf.pattern.model.Interface$1'
//begin of pattern 'org.eclipse.egf.emf.pattern.model.call.Interface.InterfacegenFeatureoverride

//end of pattern 'org.eclipse.egf.emf.pattern.model.call.Interface.InterfacegenFeatureoverride'
//begin of pattern 'org.eclipse.egf.emf.pattern.model.call.Interface.InterfacegetGenFeatureoverride

//end of pattern 'org.eclipse.egf.emf.pattern.model.call.Interface.InterfacegetGenFeatureoverride'
//begin of pattern 'org.eclipse.egf.emf.pattern.model.call.Interface.InterfacegetGenFeaturejavadocoverride

      * Returns the value of the '<em><b>Address</b></em>' attribute.
//end of pattern 'org.eclipse.egf.emf.pattern.model.call.Interface.InterfacegetGenFeaturejavadocoverride:doGenerate'
//begin of pattern 'org.eclipse.egf.emf.pattern.model.call.Interface.InterfacegetGenFeatureoverride:doGenerate

    String getAddress();

//end of pattern 'org.eclipse.egf.emf.pattern.model.call.Interface.InterfacegetGenFeatureoverride:doGenerate'
```

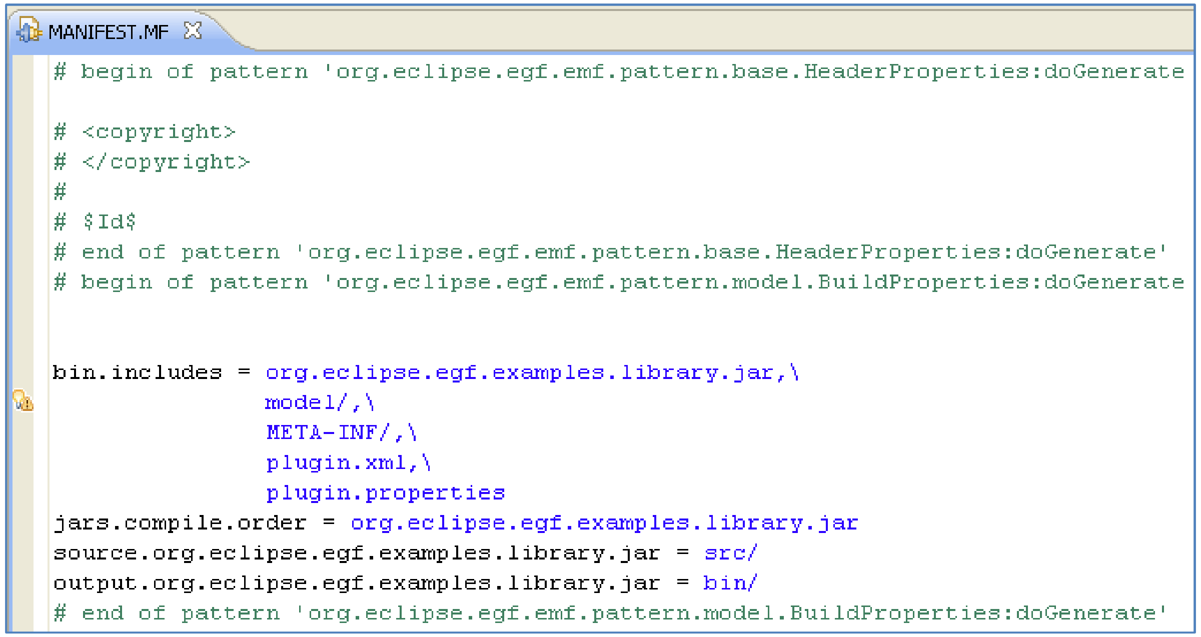
Example – EMF Generation – Build Properties

Without Trace


org.eclipse.egf.examples.library

```
# <copyright>
# </copyright>
#
# $Id$


bin.includes = org.eclipse.egf.examples.library.jar,\
               model/,\
               META-INF/,\
               plugin.xml,\
               plugin.properties
jars.compile.order = org.eclipse.egf.examples.library.jar
source.org.eclipse.egf.examples.library.jar = src/
output.org.eclipse.egf.examples.library.jar = bin/
```

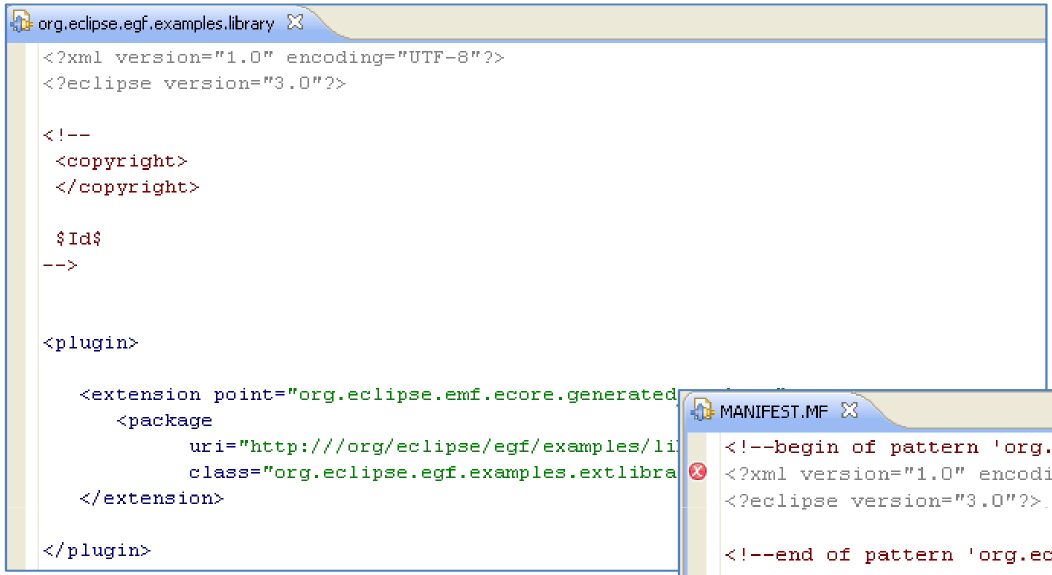
With Trace


MANIFEST.MF

```
# begin of pattern 'org.eclipse.egf.emf.pattern.base.HeaderProperties:doGenerate

# <copyright>
# </copyright>
#
# $Id$
# end of pattern 'org.eclipse.egf.emf.pattern.base.HeaderProperties:doGenerate'
# begin of pattern 'org.eclipse.egf.emf.pattern.model.BuildProperties:doGenerate


bin.includes = org.eclipse.egf.examples.library.jar,\
               model/,\
               META-INF/,\
               plugin.xml,\
               plugin.properties
jars.compile.order = org.eclipse.egf.examples.library.jar
source.org.eclipse.egf.examples.library.jar = src/
output.org.eclipse.egf.examples.library.jar = bin/
# end of pattern 'org.eclipse.egf.emf.pattern.model.BuildProperties:doGenerate'
```

Example – EMF Generation – plugin.xml

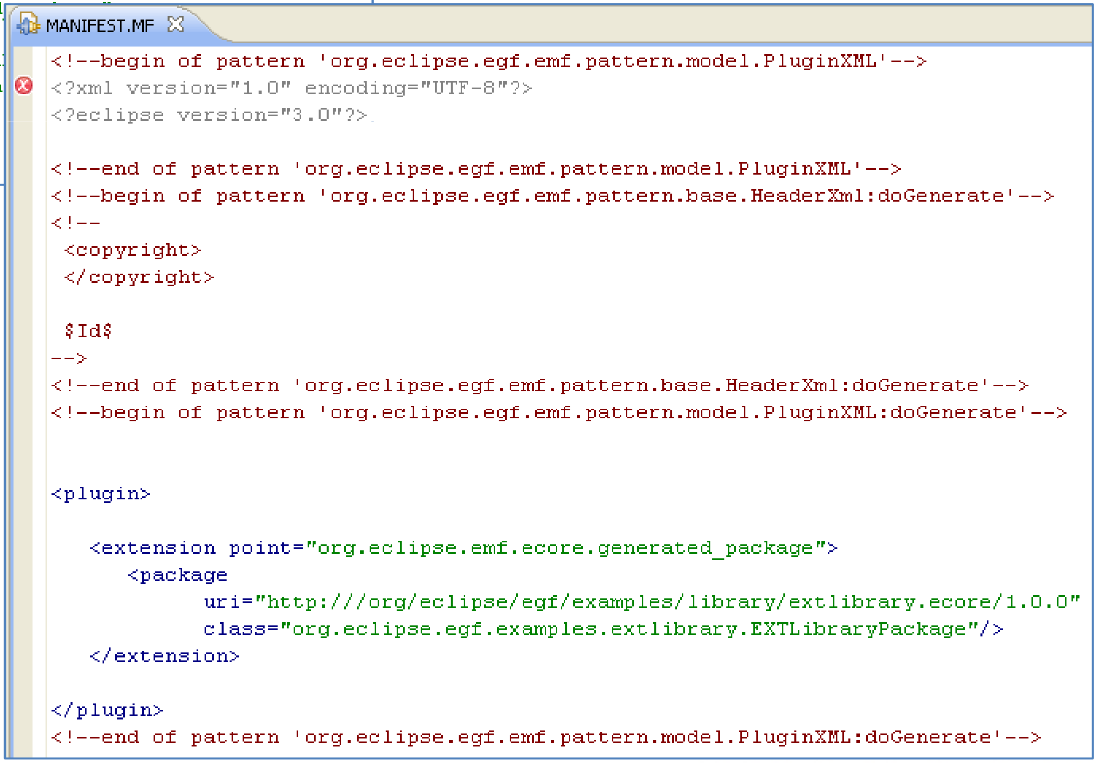
Without Trace



org.eclipse.egf.examples.library

```
<?xml version="1.0" encoding="UTF-8"?>
<?eclipse version="3.0"?>

<!--
 <copyright>
 </copyright>

 $Id$
-->


<plugin>

   <extension point="org.eclipse.emf.ecore.generated
      <package
            uri="http:///org/eclipse/egf/examples/li
            class="org.eclipse.egf.examples.extlibra
   </extension>

</plugin>
```

With Trace



MANIFEST.MF

```
<!--begin of pattern 'org.eclipse.egf.emf.pattern.model.PluginXML'-->
<?xml version="1.0" encoding="UTF-8"?>
<?eclipse version="3.0"?>

<!--end of pattern 'org.eclipse.egf.emf.pattern.model.PluginXML'-->
<!--begin of pattern 'org.eclipse.egf.emf.pattern.base.HeaderXml:doGenerate'-->
<!--
 <copyright>
 </copyright>

 $Id$
-->
<!--end of pattern 'org.eclipse.egf.emf.pattern.base.HeaderXml:doGenerate'-->
<!--begin of pattern 'org.eclipse.egf.emf.pattern.model.PluginXML:doGenerate'-->


<plugin>

   <extension point="org.eclipse.emf.ecore.generated_package">
      <package
            uri="http:///org/eclipse/egf/examples/library/extlibrary.ecore/1.0.0"
            class="org.eclipse.egf.examples.extlibrary.EXTLibraryPackage"/>
   </extension>

</plugin>
<!--end of pattern 'org.eclipse.egf.emf.pattern.model.PluginXML:doGenerate'-->
```

THALES