INRIA	Generating an ATL Execution Trace as a Traceability Weaving Model	Marcos Didonet Del Fabro, Frédéric Jouault
	A user guide	September 15th, 2006

1. Generating an ATL Execution Trace as a Traceability Weaving Model

1.1. Introduction

We use ATL to transform from a simple object model into a relational model. The ATL transformation is modified to generate traceability information.

The example is a simple transformation from Class to Relational models (*Class2Relational.atl*). The original transformation does not generate traceability information. However adding traceability information enables to know which input elements generated a given output element, and by which ATL rule as well.

The transformation *ATL2WTracer.atl* is a transformation that takes an ATL model as input and produces an ATL model that generates traceability information. The *Class2Relational.atl* transformation is transformed into *Class2Relational-WithWTracer.atl*. The transformation *Class2Relational-WithWTracer.atl* transforms Class to Relational models, and additionally generates a traceability model.

The traceability model conforms to an extension of the core weaving metamodel. A weaving model captures links between model elements. We extended the weaving metamodel with a traceability extension. The ATLAS Model Weaver prototype [2] is a tool to manipulate weaving models. In its actual implementation state, the tool needs exact information about the file paths. This information is not generated by the ATL transformation. We describe in this document how to load a traceability model in the AMW prototype.

1.2. The traceability metamodel extension

The traceability fragment below is an extension to the core weaving metamodel. The core weaving metamodel is available in the Atlantic Zoo (http://www.eclipse.org/gmt/am3/zoos/), mwcore.km3.

```
package mw_traceability {

      class TraceModel extends WModel {
            reference wovenModels[*] container subsets wovenModel :
TraceModelRef;
      }

      class TraceModelRef extends WModelRef {
            reference ownedElementRef[*] container : ElementRef oppositeOf
modelRef;
      }
---------------------------------------------------------------
    class TraceLink  extends WLink{
        attribute ruleName : String;
        reference sourceElements[*] ordered container subsets end :
WLinkEnd;
        reference targetElements[*] ordered container subsets end :
WLinkEnd;
```

```
    }

      class ElementRef extends WElementRef {
            reference modelRef: TraceModelRef oppositeOf ownedElementRef;
      }
      class TraceLinkEnd extends WLinkEnd {
      }

}
```

1.3. Traceability Model

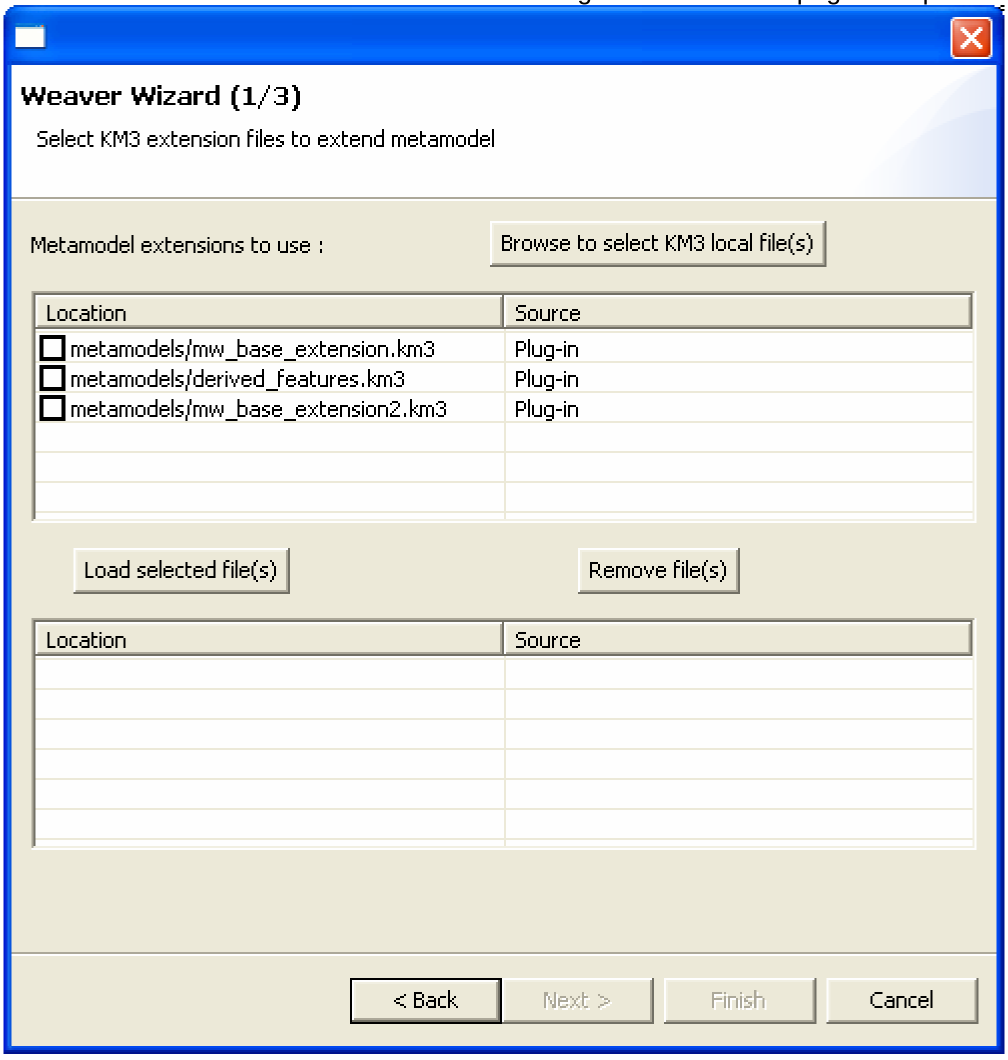
The code below shows the traceability model generated by the ATL transformation in XMI. The path information should be at lines 1 and 12. They do not have information about where the traced models are. The information that relates a referred model with his path is the **name** plus the **ref** attribute (the *ref* attribute information is missing in the model and will the automatically generated by AMW). We produce a traceability model using IN and OUT as names, but any arbitrary name may be chosen.

```
1   <wovenModels xsi:type="TraceModelRef" name="IN">
2     <ownedElementRef ref="/1"/>
3     <ownedElementRef ref="/0"/>
4     <ownedElementRef ref="/2"/>
5     <ownedElementRef ref="/3"/>
6     <ownedElementRef ref="/1/@attr.0"/>
7     <ownedElementRef ref="/0/@attr.0"/>
8     <ownedElementRef ref="/1/@attr.2"/>
9     <ownedElementRef ref="/1/@attr.1"/>
10     <ownedElementRef ref="/0/@attr.1"/>
11   </wovenModels>
12   <wovenModels xsi:type="TraceModelRef" name="OUT">
13     <ownedElementRef ref="/0"/>
14     <ownedElementRef ref="/0/@col.0"/>
15     <ownedElementRef ref="/1"/>
16     <ownedElementRef ref="/1/@col.0"/>
17     <ownedElementRef ref="/2"/>
18     <ownedElementRef ref="/3"/>
19     <ownedElementRef ref="/0/@col.1"/>
20     <ownedElementRef ref="/1/@col.1"/>
21     <ownedElementRef ref="/4"/>
22     <ownedElementRef ref="/4/@col.0"/>
23     <ownedElementRef ref="/4/@col.1"/>
24     <ownedElementRef ref="/0/@col.2"/>
25     <ownedElementRef ref="/5"/>
26     <ownedElementRef ref="/5/@col.0"/>
27     <ownedElementRef ref="/5/@col.1"/>
28   </wovenModels>
```

1.4. Loading the traceability model in AMW

We describe step-by-step how to load the modified traceability model into AMW:

In Eclipse, choose File -> New -> Model Weaver -> Weaving Model. A wizard page will open.


Weaver Wizard (1/3)

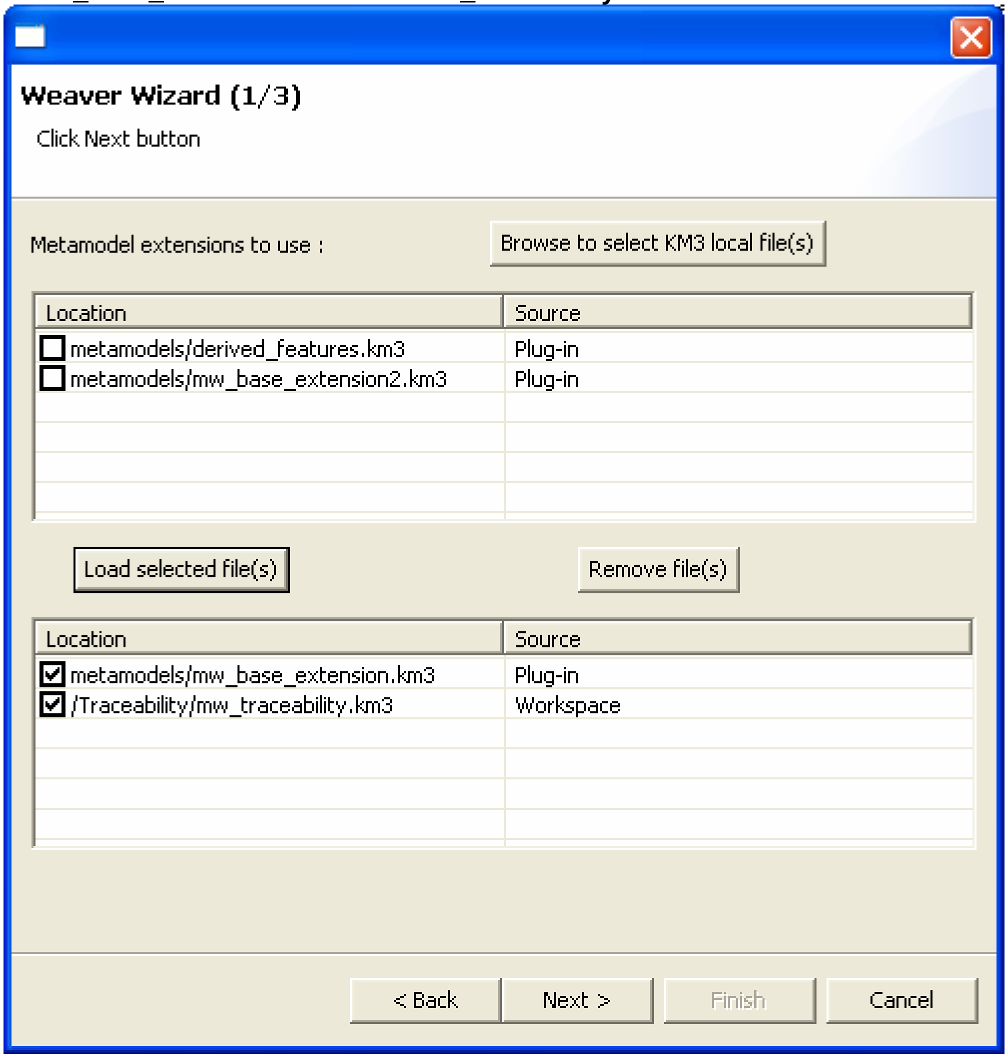
Select KM3 extension files to extend metamodel

Metamodel extensions to use :

Location	Source
☐ metamodels/mw_base_extension.km3	Plug-in
☐ metamodels/derived_features.km3	Plug-in
☐ metamodels/mw_base_extension2.km3	Plug-in

Click on *Browse to select KM3 local file(s)* and choose the **mw_traceability.km3** file from the workspace.

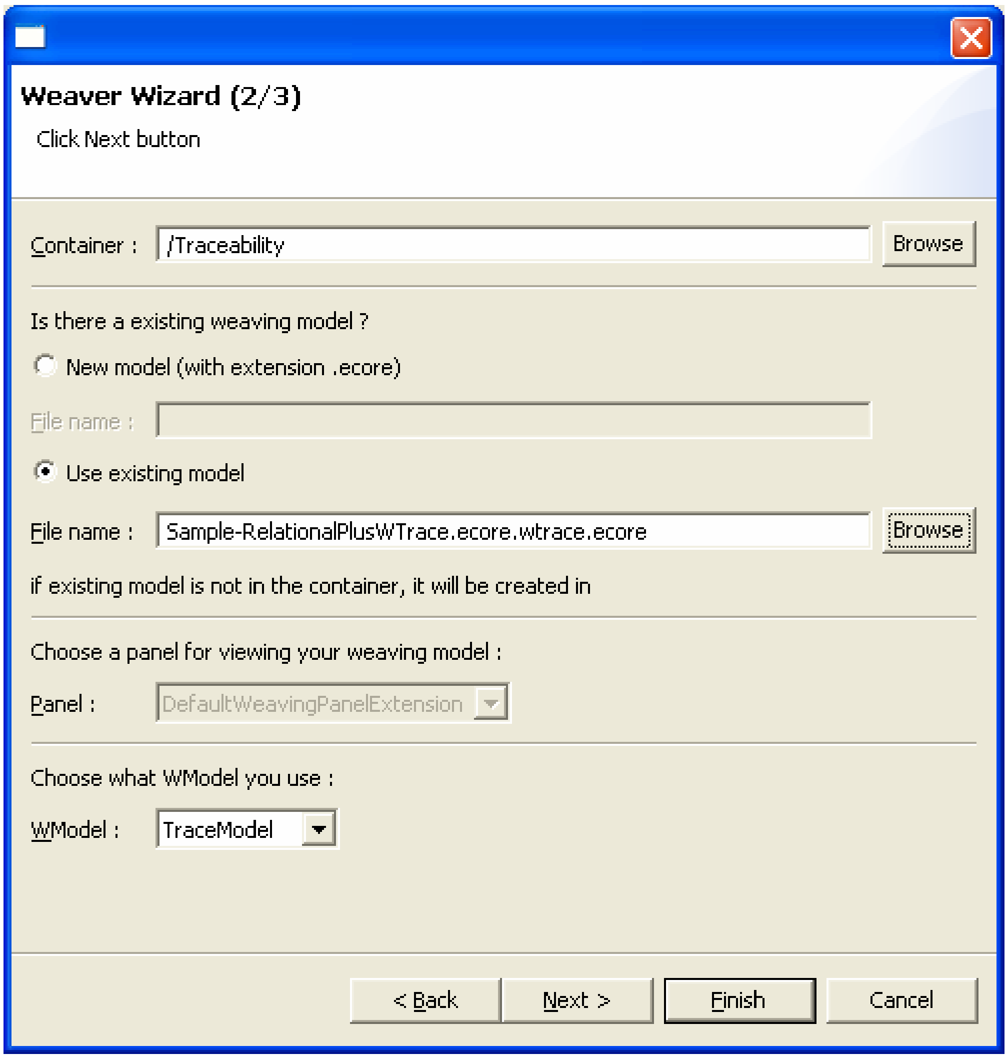
Select the **mw_base_extension.km3** and **mw_traceability.km3** and load them.



Location	Source
☐ metamodels/derived_features.km3	Plug-in
☐ metamodels/mw_base_extension2.km3	Plug-in

Location	Source
☑ metamodels/mw_base_extension.km3	Plug-in
☑ /Traceability/mw_traceability.km3	Workspace

In the second page, choose *Use Existing Model* and select the modified traceability model. In the drop-down list with the choice of WModels, choose, **TraceModel**.



Weaver Wizard (2/3)

Click Next button

Container : /Traceability

Is there a existing weaving model ?

( ) New model (with extension .ecore)

File name :

[✓ Use existing model]

File name : Sample-RelationalPlusWTrace.ecore.wtrace.ecore

if existing model is not in the container, it will be created in

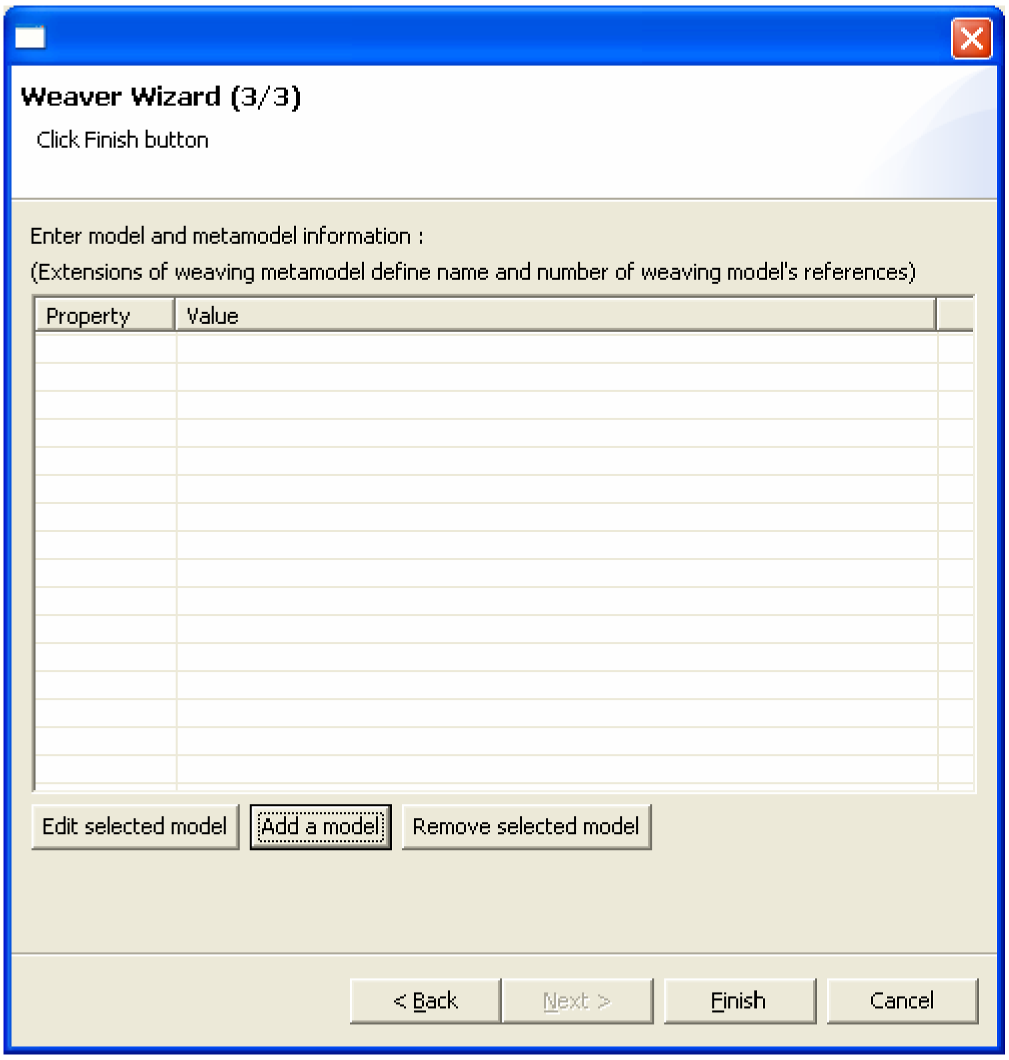
Choose a panel for viewing your weaving model :

Panel : DefaultWeavingPanelExtension

Choose what WModel you use :

WModel : TraceModel

On the next screen click on **Add a model.** In the Name input fill with the same name specified in the traceability model. In our example we put **OUT.** It must be the same name as specified in the traceability model. Choose the metamodel and model (**Relational.ecore** and **Sample-RelationalPlusWTrace.ecore Class.ecore** and **Sample-Class.ecore**). The same must be done for the **IN** model (**Relational.ecore** and **Sample-RelationalPlusWTrace.ecore**).



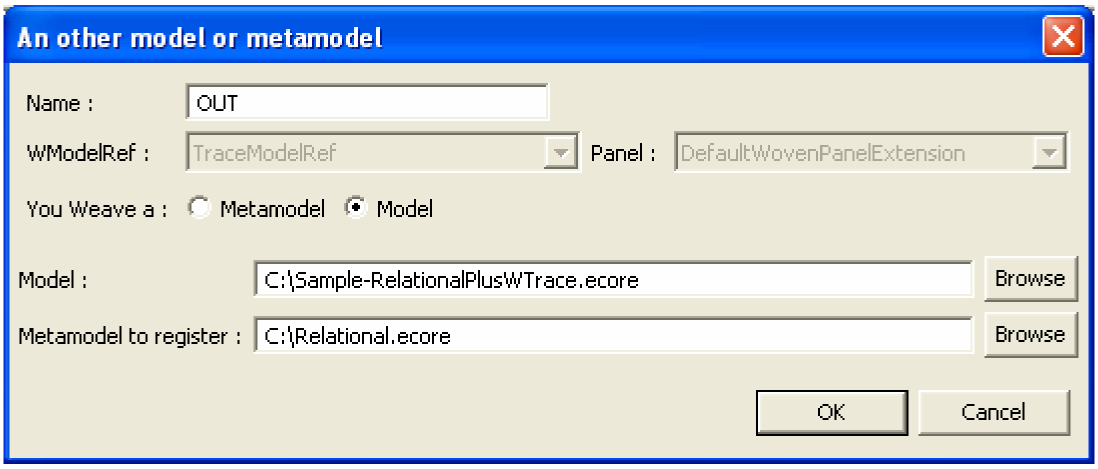
Weaver Wizard (3/3)

Click Finish button

Enter model and metamodel information :

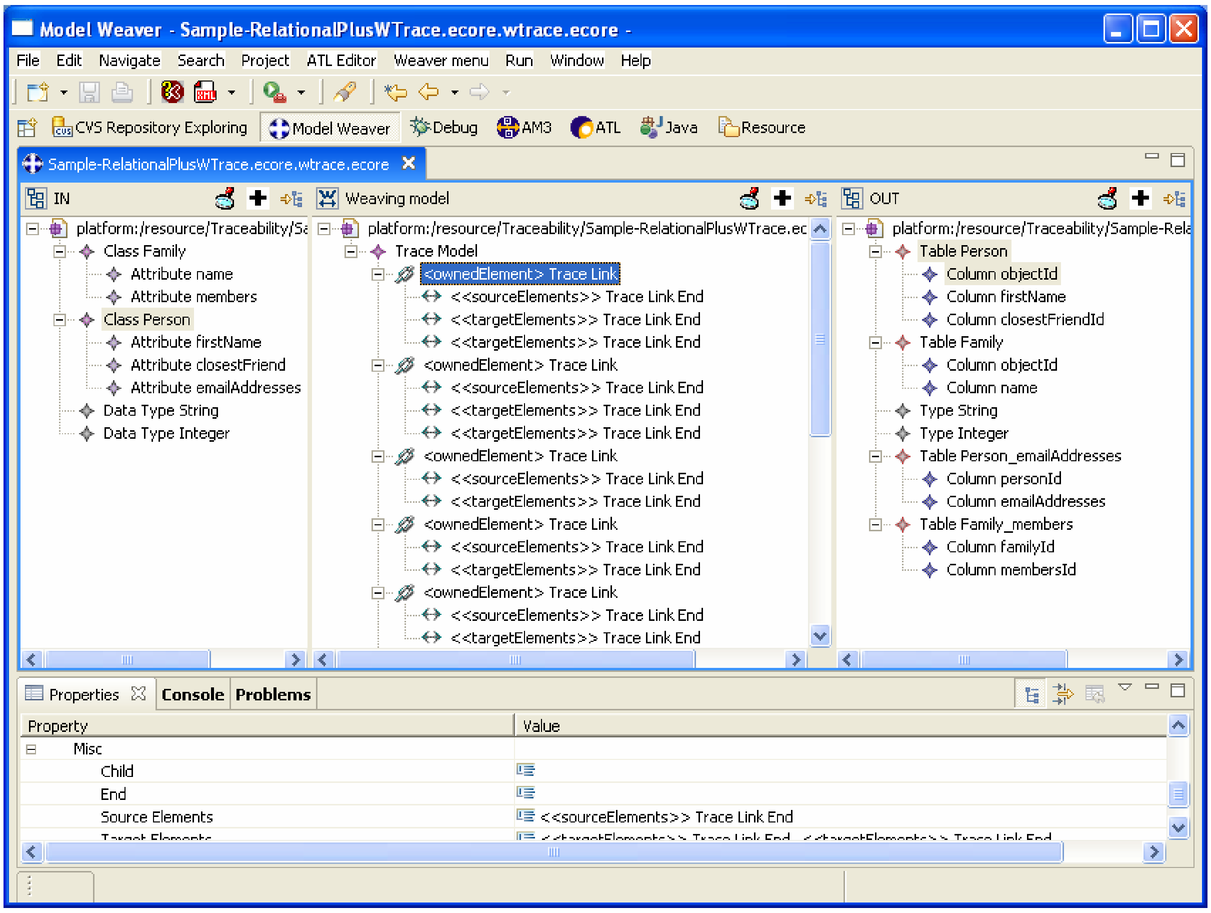
(Extensions of weaving metamodel define name and number of weaving model's references)

Property	Value



Name : OUT
WModelRef : TraceModelRef
Panel : DefaultWovenPanelExtension
You Weave a : Metamodel [✓ Model]
Model : C:\Sample-RelationalPlusWTrace.ecore
Metamodel to register : C:\Relational.ecore

The final result (the traceability model) is shown in the AMW interface.


Model Weaver - Sample-RelationalPlusWTrace.ecore.wtrace.ecore -
File Edit Navigate Search Project ATL Editor Weaver menu Run Window Help
CVS Repository Exploring
Model Weaver
Debug
AM3
ATL
Java
Resource
Sample-RelationalPlusWTrace.ecore.wtrace.ecore
IN
Weaving model
OUT
Properties
Console
Problems

```
IN
platform:/resource/Traceability/Sa
  Class Family
    Attribute name
    Attribute members
  Class Person
    Attribute firstName
    Attribute closestFriend
    Attribute emailAddresses
  Data Type String
  Data Type Integer
```

```
Weaving model
platform:/resource/Traceability/Sample-RelationalPlusWTrace.ec
  Trace Model
    <ownedElement> Trace Link
      <<sourceElements>> Trace Link End
      <<targetElements>> Trace Link End
      <<targetElements>> Trace Link End
    <ownedElement> Trace Link
      <<sourceElements>> Trace Link End
      <<targetElements>> Trace Link End
      <<targetElements>> Trace Link End
    <ownedElement> Trace Link
      <<sourceElements>> Trace Link End
      <<targetElements>> Trace Link End
    <ownedElement> Trace Link
      <<sourceElements>> Trace Link End
      <<targetElements>> Trace Link End
    <ownedElement> Trace Link
      <<sourceElements>> Trace Link End
      <<targetElements>> Trace Link End
```

```
OUT
platform:/resource/Traceability/Sample-Rela
  Table Person
    Column objectId
    Column firstName
    Column closestFriendId
  Table Family
    Column objectId
    Column name
  Type String
  Type Integer
  Table Person_emailAddresses
    Column personId
    Column emailAddresses
  Table Family_members
    Column familyId
    Column membersId
```

Property	Value
Misc	
Child	
End	
Source Elements	<<sourceElements>> Trace Link End
Target Elements	<<targetElements>> Trace Link End, <<targetElements>> Trace Link End

[1] Jouault F. Loosely Coupled Traceability for ATL. In: Proceedings of the European Conference on Model Driven Architecture (ECMDA 2005) workshop on traceability, Nuremberg, Germany

[2] ATLAS Model Weaver. http://www.eclipse.org/gmt/amw/

1.5. Version info

This example was developed using the following versions of AMW and ATL:

- AMW: July 26th 2006
- ATL: January 13th, 2006