	<b>Bridging UML profiles and Domain Specific Languages</b>	<b>ABOUZAHRA</b> <b>Anas</b>
	<b>User Guide</b>	<b>Date 28/07/2005</b>

## 1 Introduction

This document shows how to create a bridge between UML profiles and Domain Specific Languages using AMW and ATL. The general tool specification and the solution using AMW and ATL are explained in tool document (Documentation.doc). In sections 2 and 3 we explain the requirements and the steps to run the tool. Last section presents a particular example and indicates the used source files.

## 2 Requirements

- ATL (Atlas Transformation Language) and AMW (Atlas Model Weaver). Both are downloadable in the GMT web site subprojects. The ATL documentation is also available in the web site and the AMW one is downloadable as a plug-in with AMW tool. As of the AMW installed, you can find it on the eclipse Help Contents.
- The transformations MM2Profile.atl and Profile2MM.atl.


To run the tool, you have to create the domain specific metamodel and the profile model for your application. They must be conforming to the Ecore model. The profile model must conform to the profile metamodel that is provided in the example. The profile model may be created manually using the Ecore standard editor, or with another tool particularly created for that. We use our own solution to do it, which is the TCS (Textual Concrete syntax) grammar. However the TCS tools are not downloadable yet.

## 3 Using the tools

We suppose the user already have some knowledge about ATL and AMW (documentation for both tools can be found in the GMT web site). To run the tool, you have to:

- Run Eclipse version 3.1M4 or previous versions compatible with AMW and ATL.
- Create a new Weaver model conforming to the ProfileWeaver metamodel.
- Load the right metamodel as ModelRefXMLI.
- Load the left profile model as ModelRefXMLI.
- Define mapping details with weaving links (see the implementation of the mapping details in the documentation).
- Create launch configurations of each transformation: MM2Profile.atl and Profile2MM.atl.
- Run the transformations.

You will obtain two transformations which allow you to transform UML models designed with the profile to models conforming to the metamodel and vice versa.

	<b>Bridging UML profiles and Domain Specific Languages</b>	<b>ABOUZAHRA</b> <b>Anas</b>
	<b>User Guide</b>	<b>Date 28/07/2005</b>

## 4 Example

We use as example a bridge from UML profiles to MOF (as explained in the documentation). The files you need to run this example are:

- MOF metamodel ( MOF-1.4.ecore) which have to be loaded as right modelRefXMLI.
- Profile model for MOF ( profileForMof.ecore) which has to be loaded as left modelRefXMLI and for which the metamodel is the profile metamodel (Profile.ecore).
- The weaver extension which is available in the file mw\_profile\_extension.km3 or profileWeaver.ecore .
- The WprofileForMof (WprofileForMof.ecore) which is the weaving model already implemented with all links of mapping details (you can create you own one).
- The ATL metamodel (ATL-0[1].2.ecore or ATL-0.2.xmi) and the UML metamodel ( UMLDI.ecore) .

When you run the transformations (for launch configurations, see the **Erreur ! Source du renvoi introuvable.** below), you obtain the two result transformations as ATL models in XMI format or Ecore one, according to the ATL metamodel you have used: ATL-0[1].2.ecore or ATL-0.2.xmi. Then, you have to use an ATL extractor to extract textual version of the transformations.

For testing the result transformation, you have to create launch configurations of each transformation with MOF models or UML ones according to the transformation you are testing. Some models are available with source files ( Figure 2 and Figure 3 show examples of launch configurations).

For the transformation UML to MOF, the result UML model doesn't have a Model which contains all the Packages. You have to add it manually. For the XMI file for example, you have just to add the tow green lings like below:

```
<?xml version = '1.0' encoding = 'ISO-8859-1' ?>
<XMI xmi.version = '1.2' xmlns:UML = 'org.omg.xmi.namespace.UML' timestamp
= 'Fri Jul 29 09:49:50 GMT+01:00 2005'>
  <XMI.header>
    ...
  </XMI.header>
  <XMI.content>
    [redacted]
    <UML:Package xmi.id = 'a1' name = 'ATL' isSpecification = 'false'
      isLeaf = 'false' isAbstract = 'false'>
      .....
    [redacted]
  </XMI.content>
</XMI>
```

UML result models can be loaded with Poseidon UML tools for example.

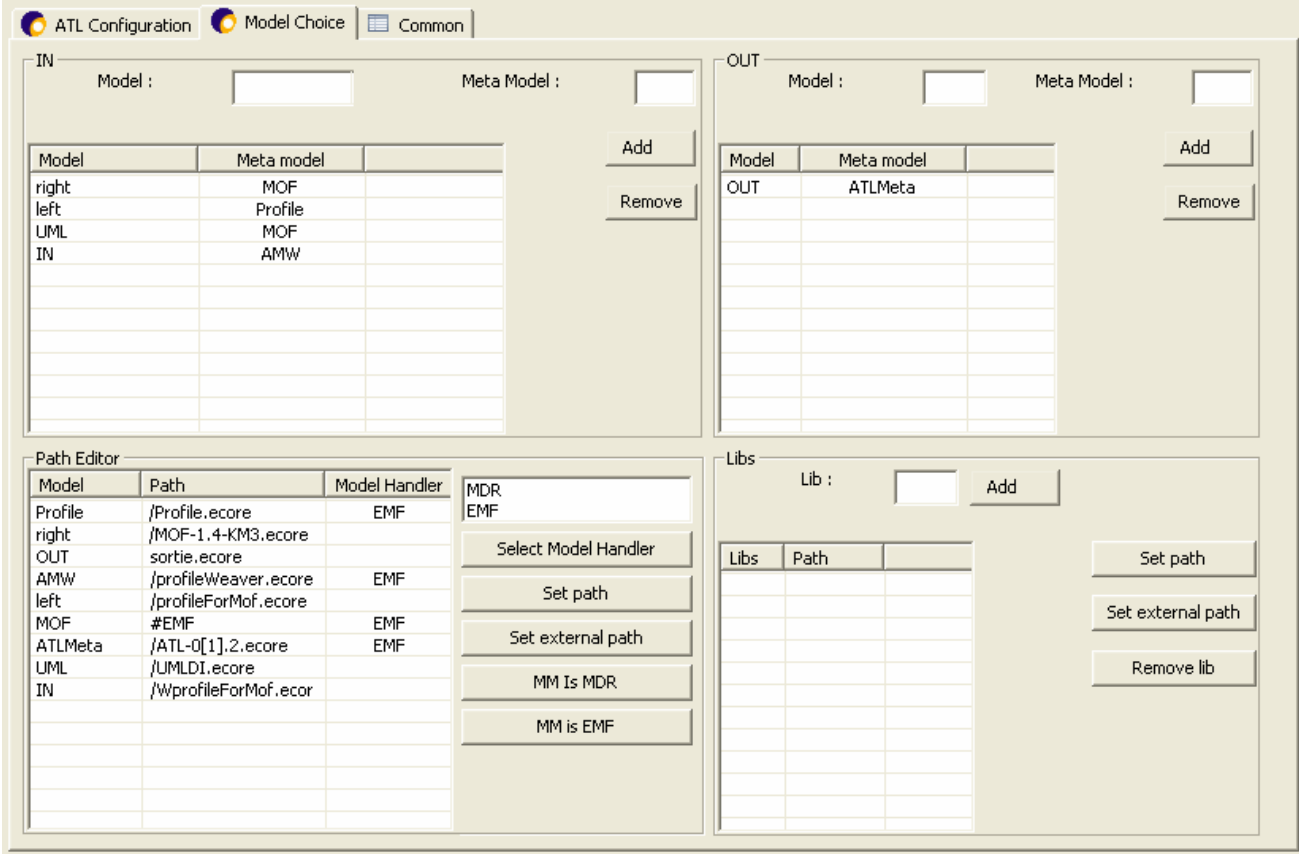
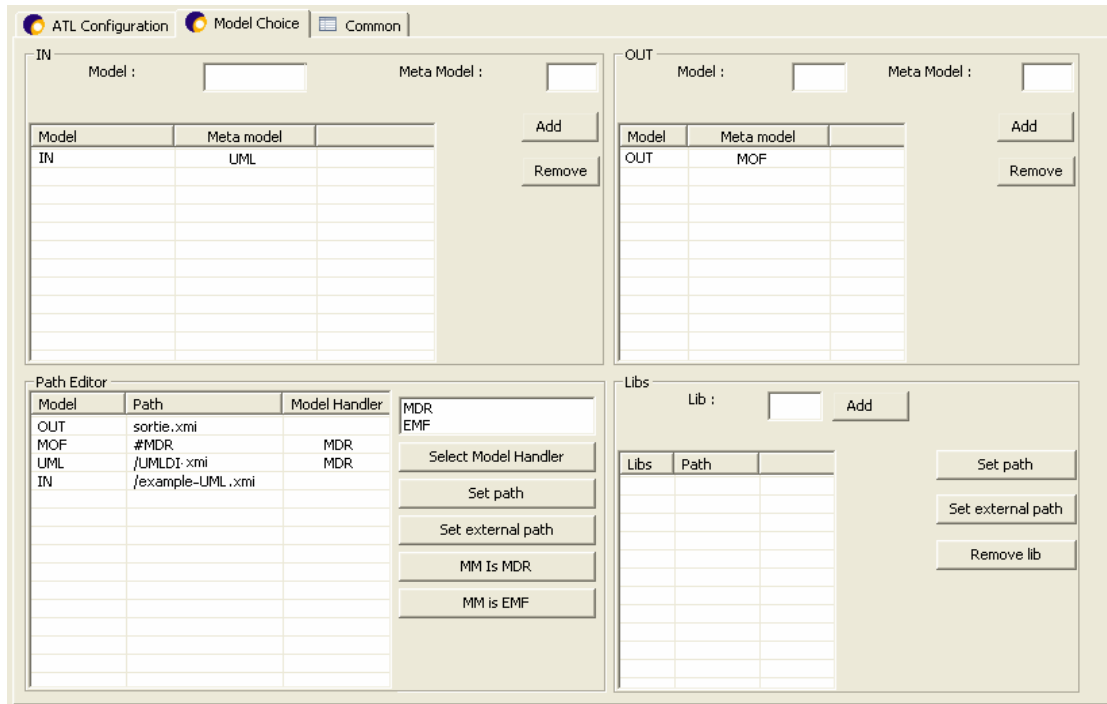
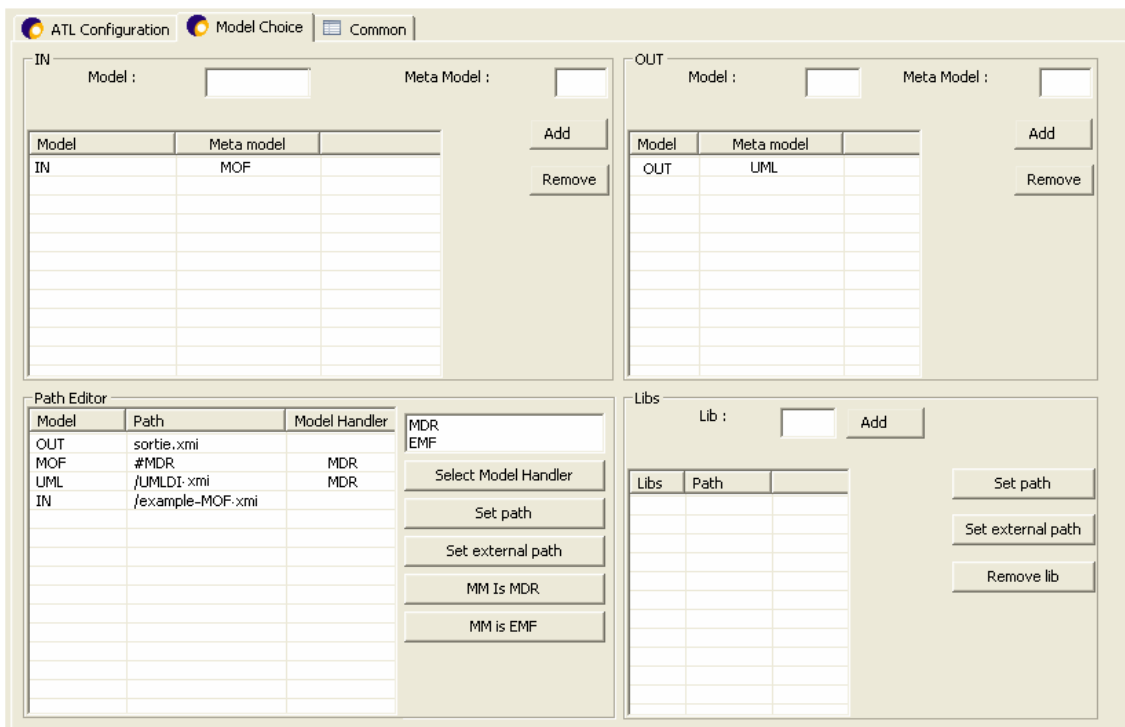



Figure 1



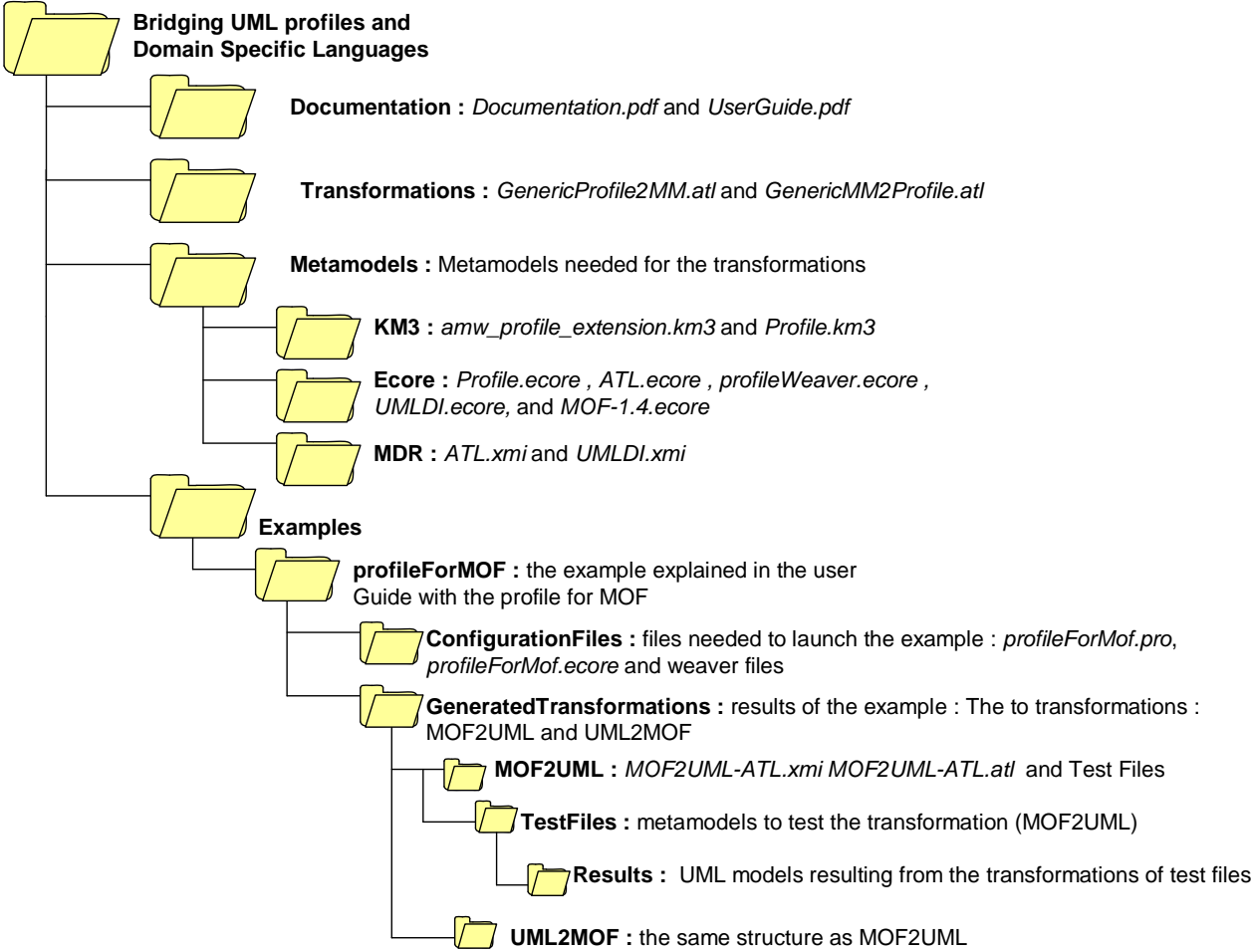
**Figure 2**



**Figure 3**

	<b>Bridging UML profiles and Domain Specific Languages</b>	<b>ABOUZAHRA</b> <b>Anas</b>
	<b>User Guide</b>	<b>Date 28/07/2005</b>

**5 Source files folder structure**



**Figure 4**