Software Languages Lab

**General Info**

- About
- Contact
- Members
- News

**Research**

- Topics
- Artifacts
- Events
- Publications [https://web.archive.org/web/20110604011804/http://www.vub.ac.be/infovoor/onderzoekers/research/team_pub.php?team_code=SOFT]
- Dissertations
- Projects
- Grants

**Education**

- Teaching
- Thesis proposals
- Project Proposals

**Internal**

- Intranet

- Index
- Recent changes

# MDE Case Studies

Modelling software is usually about abstraction. UML allows for the introduction of custom abstractions using stereotypes and/or profiles. These abstractions can be refined to a lower-level UML description using *refinement* model transformations. Several refinement transformations are provided to refine such abstractions as 'Observer', 'Singleton' and 'Applet'. Refinement transformations are also provided to refine standard UML elements into lower-level structures, such as associations to public attributes and public attributes to encapsulated attributes (with getter and setter methods).

Some case studies have been worked out using UML 1.4, UML 2 and Java. All case studies can be found in our Subversion repository [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/svn-gen/] under UML1CaseStudies [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/svn-gen/UML1CaseStudies/] and UML2CaseStudies [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/svn-gen/UML2CaseStudies/]. As part of these case studies, a number of model transformations have been written using the ATLAS Transformation Language [https://web.archive.org/web/20110604011804/http://www.eclipse.org/m2m/atl/]. If you want to try out the case studies for yourself, follow the installation instructions link below:

### Installation instructions

The following case studies are available:

- **Instant Messenger case study**
- **Breakout Game case study**

## Documentation

You can view the following presentation, including flash demo, of the instant messenger case study:

- Wagelaar, D., *MDE Case Study: Using Model Transformations for UML ands DSLs*, presentation/demo for JUGS, Zürich, Switzerland, 30 March 2006 [https://web.archive.org/web/20110604011804/http://www.jugs.ch/html/events/2006/model_engineering.html].
  [ OpenOffice [https://web.archive.org/web/20110604011804/http://

ssel.vub.ac.be/Members/DennisWagelaar/docs/uml1cs-pres.odp] ] [ Adobe PDF [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/Members/DennisWagelaar/docs/uml1cs-pres.pdf] ] [ Flash demo [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/Members/DennisWagelaar/uml1cs-demo/] (49 min.) ]

## People involved

- **Dennis Wagelaar [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/ssel/about:members:denniswagelaar]**

## Model Transformations for UML 2

Below you'll find an explanation for each of the model transformations based on the UML 2 meta-model. All transformations use an 'IN' and an 'OUT' UML2 model next to any specified models, except where specified differently.

### UML2Copy

This is a basic transformation that copies all UML2 model elements from the input model to the output model. It includes a condition that makes sure only the elements from the 'IN' model are copied.

| | |
|---|---|
| **UML2Copy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2Copy.atl?view=markup]** | Exhaustive copy of a UML2 model based on auto-generated, explicit copying rules for each meta-class. Does not copy stereotype applications. |
| Meta-models: | UML2 = UML2-by-URI built-in meta-model or UML2Profiles.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/metamodels/UML2Profiles.ecore?view=markup] |

### UML2ProfileCopy

This is a transformation that copies particular kinds of stereotype applications from the input model to the output model. It is not a stand-alone transformation, but is meant to be superimposed on top of the UML2Copy transformation. Note that it is possible to superimpose any number of transformations on top of eachother and this transformation is usually the last on the stack.

| | |
|---|---|
| **UML2ProfileCopy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2ProfileCopy.atl?view=markup]** | Exhaustive copy of particular stereotype applications based on auto-generated, explicit copying rules for each stereotype meta-class. |
| Meta-models: | UML2 = UML2Profiles.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/ |

| | |
|---|---|
| | metamodels/UML2Profiles.ecore?view=markup] |
| Superimposed on: | UML2Copy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2Copy.atl?view=markup] |

**Note:** The UML2Profiles.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/metamodels/UML2Profiles.ecore?view=markup] meta-model is a special "wrapper" meta-model that imports the standard UML meta-model as well as all applicable UML profiles. UML profiles can be treated as Ecore meta-models, since they include EClass (meta-class) representations of each Stereotype.

### UML2Profiles

This transformation applies the Accessors [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/profiles/Accessors.profile.uml?view=markup] profile, if necessary. It is not a stand-alone transformation, but is meant to be superimposed on top of the UML2Copy transformation and superimposed by the UML2ProfileCopy transformation.

| **UML2Profiles.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2Profiles.atl?view=markup]** | Applies all necessary profiles. |
|---|---|
| Meta-models: | UML2 = UML2Profiles.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/metamodels/UML2Profiles.ecore?view=markup] |
| Models: | ACCESSORS = Accessors.profile.uml [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/profiles/Accessors.profile.uml?view=markup] |
| Superimposed on: | UML2Copy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2Copy.atl?view=markup] |
| Superimposed by: | UML2ProfileCopy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2ProfileCopy.atl?view=markup] |

### UML2Accessors

This transformation introduces accessor methods (getters and setters) for all public properties and then converts those properties to private properties. It is not a stand-alone transformation, but is meant to be superimposed on top of the UML2Copy transformation and superimposed by the UML2ProfileCopy transformation.

| | |
|---|---|
| **UML2Accessors.atl [https://** | Introduces accessor operations for all public |

| | |
|---|---|
| **web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2Accessors.atl?view=markup]** | properties. |
| Meta-models: | UML2 = UML2Profiles.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/metamodels/UML2Profiles.ecore?view=markup] |
| Models: | ACCESSORS = Accessors.profile.uml [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/profiles/Accessors.profile.uml?view=markup]<br>UML2TYPES = PrimitiveTypes.uml [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/models/PrimitiveTypes.uml?view=markup] |
| Libraries: | Strings = Strings.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Strings.atl?view=markup]<br>UML2 = UML2.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/UML2.atl?view=markup]<br>Mappings = JavaMappings.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/JavaMappings.atl?view=markup] (or other library implementing the same helpers)<br>Java = Java1.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Java1.atl?view=markup] or Java2.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Java2.atl?view=markup] (depending on desired datatype mapping and only when mapping to java) |
| Superimposed on: | UML2Copy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2Copy.atl?view=markup] |
| Superimposed by: | UML2ProfileCopy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2ProfileCopy.atl?view=markup] |
| Depends on: | UML2Profiles |

### UML2Observer, UML2JavaObserver

These transformations implement the Observer [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/profiles/Observer.profile.uml?view=markup] profile. They introduce the infrastructure to implement the “Observer”, “Observable” and “subscribe” stereotypes. Any instances of “Observer” classes that have a “subscribe” association to an “Observable” class, will automatically be subscribed to the “Observable” instance that they are linked to. In addition, all setter operations of the “Observable” Class will be updated to do a notify as well. An “update” operation is introduced for each “Observer” class,

which is configured to invoke any “onXYZChange” operations, where “XYZ” is the name of the property that is changed. They are not stand-alone transformations, but are meant to be superimposed on top of the UML2Copy transformation and superimposed by the UML2ProfileCopy transformation. There are two variants available:

| **UML2Observer.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2Observer.atl?view=markup]** | Implements the Observer profile by introducing observer pattern infrastructure. |
|---|---|
| Meta-models: | UML2 = UML2Profiles.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/metamodels/UML2Profiles.ecore?view=markup] |
| Models: | OBSERVER = Observer.uml [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/models/Observer.uml?view=markup]<br>UML2TYPES = PrimitiveTypes.uml [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/models/PrimitiveTypes.uml?view=markup] |
| Libraries: | Strings = Strings.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Strings.atl?view=markup]<br>UML2 = UML2.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/UML2.atl?view=markup]<br>Mappings = JavaMappings.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/JavaMappings.atl?view=markup] (or other library implementing the same helpers)<br>Java = Java1.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Java1.atl?view=markup] or Java2.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Java2.atl?view=markup] (depending on desired datatype mapping and only when mapping to java) |
| Superimposed on: | UML2Copy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2Copy.atl?view=markup] |
| Superimposed by: | UML2ProfileCopy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2ProfileCopy.atl?view=markup] |
| Depends on: | UML2Accessors |
| **UML2JavaObserver.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2JavaObserver.atl?** | Implements the Observer profile using the `java.util.Observer` API. |

| **view=markup]** | |
|---|---|
| Meta-models: | UML2 = UML2Profiles.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/metamodels/UML2Profiles.ecore?view=markup] |
| Models: | IMPLTYPES = JavaTypes.uml [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/models/JavaTypes.uml?view=markup]<br>UML2TYPES = PrimitiveTypes.uml [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/models/PrimitiveTypes.uml?view=markup] |
| Libraries: | Strings = Strings.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Strings.atl?view=markup]<br>UML2 = UML2.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/UML2.atl?view=markup]<br>Mappings = JavaMappings.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/JavaMappings.atl?view=markup] (or other library implementing the same helpers)<br>Java = Java1.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Java1.atl?view=markup] or Java2.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Java2.atl?view=markup] (depending on desired datatype mapping and only when mapping to java) |
| Superimposed on: | UML2Copy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2Copy.atl?view=markup] |
| Superimposed by: | UML2ProfileCopy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2ProfileCopy.atl?view=markup] |
| Depends on: | UML2Accessors |

### UML2AbstractFactory

This transformation implements the AbstractFactory [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/profiles/AbstractFactory.profile.uml?view=markup] profile. It is not a stand-alone transformation, but is meant to be superimposed on top of the UML2Copy transformation and superimposed by the UML2ProfileCopy transformation.

| **UML2AbstractFactory.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2AbstractFactory.atl?view=markup]** | Implements the AbstractFactory profile. |
|---|---|

| | |
|---|---|
| Meta-models: | UML2 = UML2Profiles.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/metamodels/UML2Profiles.ecore?view=markup] |
| Libraries: | UML2 = UML2.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/UML2.atl?view=markup]<br>Mappings = JavaMappings.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/JavaMappings.atl?view=markup] (or other library implementing the same helpers)<br>Java = Java1.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Java1.atl?view=markup] or Java2.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Java2.atl?view=markup] (depending on desired datatype mapping and only when mapping to java) |
| Superimposed on: | UML2Copy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2Copy.atl?view=markup] |
| Superimposed by: | UML2ProfileCopy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2ProfileCopy.atl?view=markup] |

### UML2Singleton

This transformation implements the Singleton [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/profiles/Singleton.profile.uml?view=markup] profile. It is not a stand-alone transformation, but is meant to be superimposed on top of the UML2Copy transformation and superimposed by the UML2ProfileCopy transformation.

| | |
|---|---|
| **UML2Singleton.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2Singleton.atl?view=markup]** | Implements the Singleton profile. |
| Meta-models: | UML2 = UML2Profiles.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/metamodels/UML2Profiles.ecore?view=markup] |
| Libraries: | UML2 = UML2.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/UML2.atl?view=markup]<br>Mappings = JavaMappings.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/JavaMappings.atl?view=markup] (or other library implementing the same helpers)<br>Java = Java1.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/ |

| | |
|---|---|
| | viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Java1.atl?view=markup] or Java2.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Java2.atl?view=markup] (depending on desired datatype mapping and only when mapping to java) |
| Superimposed on: | UML2Copy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2Copy.atl?view=markup] |
| Superimposed by: | UML2ProfileCopy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2ProfileCopy.atl?view=markup] |

### UML2Applet, UML2MIDlet

These transformations implement the Applet [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/profiles/Applet.profile.uml?view=markup] profile. They are not stand-alone transformations, but are meant to be superimposed on top of the UML2Copy transformation and superimposed by the UML2ProfileCopy transformation. There are two variants available:

| **UML2Applet.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2Applet.atl?view=markup]** | Implements the Applet profile using a Java AWT Applet. |
|---|---|
| Meta-models: | UML2 = UML2Profiles.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/metamodels/UML2Profiles.ecore?view=markup] |
| Models: | IMPLTYPES = JavaTypes.uml [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/models/JavaTypes.uml?view=markup] |
| Libraries: | UML2 = UML2.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/UML2.atl?view=markup]<br>Mappings = JavaMappings.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/JavaMappings.atl?view=markup] (or other library implementing the same helpers)<br>Java = Java1.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Java1.atl?view=markup] or Java2.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Java2.atl?view=markup] (depending on desired datatype mapping and only when mapping to java) |
| Superimposed on: | UML2Copy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2Copy.atl?view=markup] |

| | |
|---|---|
| Superimposed by: | UML2ProfileCopy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2ProfileCopy.atl?view=markup] |
| **UML2MIDlet.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2MIDlet.atl?view=markup]** | Implements the Applet profile using a Java MIDP MIDlet. |
| Meta-models: | UML2 = UML2Profiles.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/metamodels/UML2Profiles.ecore?view=markup] |
| Models: | MIDLET = MIDlet.uml [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/models/MIDlet.uml?view=markup] |
| Libraries: | Strings = Strings.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Strings.atl?view=markup]<br>UML2 = UML2.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/UML2.atl?view=markup]<br>Mappings = JavaMappings.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/JavaMappings.atl?view=markup] (or other library implementing the same helpers)<br>Java = Java1.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Java1.atl?view=markup] or Java2.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Java2.atl?view=markup] (depending on desired datatype mapping and only when mapping to java) |
| Superimposed on: | UML2Copy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2Copy.atl?view=markup] |
| Superimposed by: | UML2ProfileCopy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2ProfileCopy.atl?view=markup] |

### UML2AsyncMethods

This transformation implements the Async [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/profiles/Async.profile.uml?view=markup] profile. It wraps the execution of all asynchronous operations in a thread. It is not a stand-alone transformation, but is meant to be superimposed on top of the UML2Copy transformation and superimposed by the UML2ProfileCopy transformation.

| | |
|---|---|
| **UML2AsyncMethods.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/** | Implements the Async profile. |

| **UML2AsyncMethods.atl?view=markup]** | |
|---|---|
| Meta-models: | UML2 = UML2Profiles.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/metamodels/UML2Profiles.ecore?view=markup] |
| Libraries: | Mappings = JavaMappings.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/JavaMappings.atl?view=markup] (or other library implementing the same helpers)<br>Java = Java1.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Java1.atl?view=markup] or Java2.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Java2.atl?view=markup] (depending on desired datatype mapping and only when mapping to java) |
| Superimposed on: | UML2Copy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2Copy.atl?view=markup] |
| Superimposed by: | UML2ProfileCopy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2ProfileCopy.atl?view=markup] |

### UML2DataTypes

This transformation translates all UML2-specific data types to programming language types. As such, it must be applied after all transformations that introduce references to UML2-specific data types. It is not a stand-alone transformation, but is meant to be superimposed on top of the UML2Copy transformation and superimposed by the UML2ProfileCopy transformation.

| **UML2DataTypes.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2DataTypes.atl?view=markup]** | Transforms all UML2-specific data types to programming language types. |
|---|---|
| Meta-models: | UML2 = UML2Profiles.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/metamodels/UML2Profiles.ecore?view=markup] |
| Models: | IMPLTYPES = JavaTypes.uml [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/models/JavaTypes.uml?view=markup] (or any other model containing the primitive types needed by the Mappings library) |
| Libraries: | Strings = Strings.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Strings.atl?view=markup]<br>UML2 = UML2.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/UML2.atl?view=markup] |

| | |
|---|---|
| | Mappings = JavaMappings.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/JavaMappings.atl?view=markup] (or other library implementing the same helpers)<br>Java = Java1.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Java1.atl?view=markup] or Java2.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Java2.atl?view=markup] (depending on desired datatype mapping and only when mapping to java) |
| Superimposed on: | UML2Copy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2Copy.atl?view=markup] |
| Superimposed by: | UML2ProfileCopy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2ProfileCopy.atl?view=markup] |
| Depends on: | All transformations that introduce references to UML2-specific data types |

### UML2ToJava

This transformation generates Java code from a UML2 model. As such, it is applied after all UML2-to-UML2 refinement transformations. It requires a 'parameters' model that specifies the 'path' to write the code to. This 'parameters' model is an XML file that needs to be injected using AM3 Ant tasks [https://web.archive.org/web/20110604011804/http://wiki.eclipse.org/index.php/ATL_Howtos#Using_an_XML_file].

| **UML2ToJava.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/UML2ToJava.atl?view=markup]** | Generates Java code from a UML2 model. |
|---|---|
| Meta-models: | UML2 = UML2Profiles.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/metamodels/UML2Profiles.ecore?view=markup]<br>XML = XML.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/metamodels/XML.ecore?view=markup] |
| Models: | parameters = parameters.xml [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-instantmessenger-model/outmodels/test/parameters.xml?view=markup] (this is an example 'parameters' model that must be injected using AM3 Ant tasks) |
| Libraries: | Strings = Strings.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Strings.atl?view=markup]<br>UML2 = UML2.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/ |

| | |
|---|---|
| | viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/UML2.atl?view=markup] |

### EModelCopyGenerator

This transformation generates an ATL transformation that copies models using the given Ecore meta-model. It has been used to generate parts of the UML2Copy and UML2ProfileCopy transformations. It requires a 'config.atl' library that specifies 'path', 'inclusionCondition' and 'useQualifiedName' helpers.

| **EModelCopyGenerator.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/EModelCopyGenerator.atl?view=markup]** | Generates an ATL transformation that copies models using the given Ecore meta-model. |
|---|---|
| Meta-models: | ECORE = Ecore-by-URI built-in meta-model or built-in #MOF meta-model |
| Models: | IN = UML.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/metamodels/UML.ecore?view=markup] (or any Ecore meta-model or UML2 profile you want to create copying rules for) |
| Libraries: | Config = Config.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/lib/Config.atl?view=markup] |

### ConfigToBuildFile

This transformation generates an Ant 'build.xml' file from a configuration model. This 'build.xml' file is an Ant template that contains macros and targets for executing the 'UML2…' transformations according to the given configuration. The 'build.xml' file must be extracted using AM3 Ant tasks [https://web.archive.org/web/20110604011804/http://wiki.eclipse.org/index.php/AM3_Ant_Tasks#am3.saveModel].

| **ConfigToBuildFile.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/ConfigToBuildFile.atl?view=markup]** | Generates a build.xml file from a Transformations.ecore config file. |
|---|---|
| Meta-models: | CFG = Transformations.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/configuration/Transformations.ecore?view=markup]<br>XML = XML.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/metamodels/XML.ecore?view=markup] |
| Models: | IN = a configuration model such as config.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-instantmessenger-default/config.ecore?view=markup] |

| | |
|---|---|
| | OUT = the output 'build.xml' file |

ConfigToParameters

This transformation generates a 'parameters.xml' file from a configuration model. This 'parameters.xml' file is used by the UML2ToJava transformation. The 'parameters.xml' file must be extracted using AM3 Ant tasks [https://web.archive.org/web/20110604011804/http://wiki.eclipse.org/index.php/AM3_Ant_Tasks#am3.saveModel].

| | |
|---|---|
| **ConfigToParameters.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/ConfigToParameters.atl?view=markup]** | Generates a parameters.xml file from a Transformations.ecore config file. |
| Meta-models: | CFG = Transformations.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/configuration/Transformations.ecore?view=markup]<br>XML = XML.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-transformations/metamodels/XML.ecore?view=markup] |
| Models: | IN = a configuration model such as config.ecore [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/UML2CaseStudies/uml2cs-instantmessenger-default/config.ecore?view=markup]<br>OUT = the output 'parameters.xml' file |

## Model Transformations for UML 1.4

Below you'll find an explanation for each of the model transformations based on the UML 1.4 meta-model:

CopyModel, ModelCopy

These are very simple transformations, meant to copy all elements of one model into another model. There are two variants available:

| | |
|---|---|
| **CopyModel.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/CopyModel.atl]** | Deep copy of a UML model based on refinement mode. This means that copying rules are only required for the root model elements. Also, the input and output model are required to have the same meta-model (and repository), since this is required by refinement mode. |
| **ModelCopy.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/ModelCopy.atl]** | Exhaustive copy of a UML model based on auto-generated, explicit copying rules for each meta-class. Different (yet compatible) meta-models can be used for input and output model. Also, different repositories can be used, e.g. for copying a model from MDR to EMF. |

Note that the ModelCopy.atl [https://web.archive.org/web/20110604011804/

http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/ModelCopy.atl] transformation has been generated with the ModelCopyGenerator.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/ModelCopyGenerator.atl] transformation. The "inclusionCondition" helper used in the ModelCopyGenerator.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/ModelCopyGenerator.atl] transformation is in turn derived from the output of the ModelUsage.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/ModelUsage.atl] query transformation, which is generated by the ModelUsageGenerator.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/ModelUsageGenerator.atl] transformation 8-O.

### MergeModel, ModelMerge

These transformations are meant to merge a model with the main model. The MERGE model is merged with the IN model and written to the OUT model. They are currently limited to copying elements that can occur in UML Class Diagrams (including Action Semantics). There are two variants available:

| | |
|---|---|
| **MergeModel.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/MergeModel.atl]** | Based on refinement mode. This means that all models are required to have the same meta-model (and repository), since this is required by refinement mode. Partly optimised for speed by caching intermediate results in OCL helper attributes. |
| **ModelMerge.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/ModelMerge.atl]** | Based on auto-generated, explicit copying rules for each meta-class. Different (yet compatible) meta-models (and repositories) can be used for the models. **NOTE:** This transformation is no longer maintained/used because it is too complex/inefficient compared to its alternative. |

Note that most other refinement transformations are derivatives (and include the rules) of the MergeModel.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/MergeModel.atl] transformation. This allows the refinement transformations to include static model content into the target model without having to generate it (and use complex queries to make links to this newly generated content).

### AssociationAttributes, Java2AssociationAttributes

These transformations introduce Attributes (with Java initial values) for each Association End. There are two variants available:

| | |
|---|---|
| **AssociationAttributes.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/** | Uses strict Java 1 API. Works on J2ME |

| | |
|---|---|
| ***checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/AssociationAttributes.atl]** | MIDP 1.0 as well. |
| **Java2AssociationAttributes.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/Java2AssociationAttributes.atl]** | Uses the Java 2 Collections framework. Works on J2ME PP 1.0 as well. |

Accessors, Java2Accessors

These transformations introduce accessor Operations/Methods (with Java bodies) for each public Attribute. The public Attributes are then made private. This transformation is also used in refactoring as “Encapsulate Field”. There are two variants available:

| | |
|---|---|
| **Accessors.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/Accessors.atl]** | Uses strict Java 1 API. Works on J2ME MIDP 1.0 as well. |
| **Java2Accessors.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/Java2Accessors.atl]** | Uses the Java 2 Collections framework. Works on J2ME PP 1.0 as well. |

Observer, JavaObserver

These transformations introduce the Java infrastructure to implement the “Observer”, “Observable” and “subscribe” Stereotypes. Any instances of “Observer” Classes that have a “subscribe” Association to an “Observable” Class, will automatically be subscribed to the “Observable” instance that they are linked to. In addition, all setter Methods of the “Observable” Class will be updated to do a notify as well. An “update” Operation/Method will be introduced for each “Observer” Class, which is configured to invoke any “onXYZChange” Operations, where “XYZ” is the name of the Attribute that is changed. There are two variants available:

| | |
|---|---|
| **Observer.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/Observer.atl]** | Uses strict Java 1 API. Works on J2ME MIDP 1.0 as well. |
| **JavaObserver.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/JavaObserver.atl]** | Uses the java.util.Observer framework. Also uses Java reflection to detect any “onXYZChange” handler methods. This means that “onXYZChange” methods of subclasses will also be detected and without a recompile of the superclass. |

Singleton

A simple transformation that adds a static “getInstance” Operation/ Method and a static “instance” Attribute to each Class with a “Singleton” Stereotype:

| | |
|---|---|
| **Singleton.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/Singleton.atl]** | Does not prevent direct usage of the constructor. Uses strict Java 1 API. Works on J2ME MIDP 1.0 as well. |

Applet, MIDlet

These transformations introduce Java Applet infrastructure for each Class with an “Applet” stereotype. There are two variants available:

| | |
|---|---|
| **Applet.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/Applet.atl]** | Turns each “Applet” Class into a java.applet.Applet and introduces the “init”, “getParameterInfo” and “getAppletInfo” Operations/Methods. |
| **MIDlet.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/MIDlet.atl]** | Turns each “Applet” Class into a javax.microedition.midlet.MIDlet and introduces the “startApp”, “pauseApp”, “destroyApp”, “init”, “destroy”, “start” and “stop” Operations/Methods. Works on J2ME MIDP 1.0 or up. |

### DataTypes, Java2DataTypes

These transformations replace the OCL types with Java types. There are two variants available:

| | |
|---|---|
| **DataTypes.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/DataTypes.atl]** | Uses strict Java 1 API. Works on J2ME MIDP 1.0 as well. |
| **Java2DataTypes.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/Java2DataTypes.atl]** | Uses the Java 2 Collections framework. Works on J2ME PP 1.0 as well. |

### AsyncMethods

A simple transformation that wraps the Method body of each Operation with an “asynchronous” Stereotype inside a Thread:

| | |
|---|---|
| **AsyncMethods.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/AsyncMethods.atl]** | Uses java.lang.Thread, which is part of the Java 1.0 API. Works on J2ME MIDP 1.0 as well. |

### UMLtoJava

An ATL query that generates Java code from a low-level (i.e. fully refined) UML model. Note that no code is generated for Associations, since the AssociationAttributes transformation is responsible for translating them to implementation-level attributes.

| | |
|---|---|
| **UMLtoJava.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/svn-gen/UML1CaseStudies/uml1cs-transformations/UMLtoJava.atl]** | Requires a “config.atl” library providing the helper methods `path()`, which prepends the filesystem path for generated output, and `ignore()`, which indicates what element names not to generate code for. See also config.atl [https://web.archive.org/web/20110604011804/http://ssel.vub.ac.be/viewcvs/viewcvs.py/*checkout*/UML1CaseStudies/uml1cs-instantmessenger-model/outmodels/test2mm/config.atl] for an example configuration. |