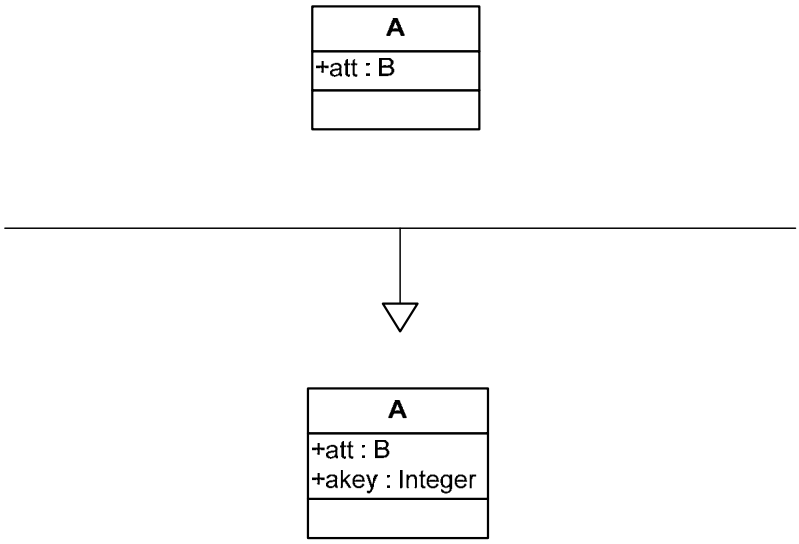
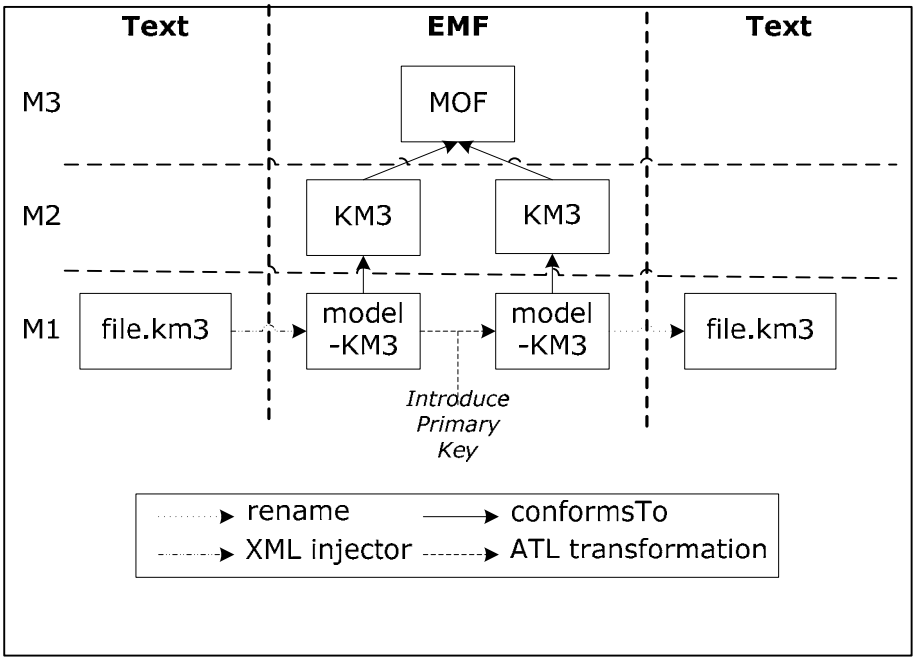
INRIA	**ATL Transformation** **Catalogue of Model Transformations**	**Author** **Baudry Julien** **Jul.baudry <*at*> gmail.com**
	Documentation	Aug 7th 2006

1. ATL Transformation Example: introduce primary key

This example is extract from Catalogue of Model Transformations by K. Lano.
Section 1.6: Introduce primary key, page 6.





2. ATL Transformation overview

2.1. Description

This transformation applies to any persistent class. If the class does not already have a primary key, it introduces a new identity attribute, of integer type.

2.2. Purpose

This is an essential step for implementation of a data model in relational database.

2.3. Rules specification

Our transformation has the same source and the target metamodel, KM3. We use 2 different names (KM3 and KM3target), but they refer to the same metamodel.

- For a Metamodel element, another Metamodel element is created :
  - with the same name and location,
  - Linked to the same contents.

- For a Package element, another Package element is created :
  - with the same name,
  - Linked to the same contents.

- For a class element, another Class element is created :
  - We create another class with the same name,
  - Abstract if the source class is abstract,
  - linked the same structural feature,
  - with an new attribute named <name of the class>+'key', the key of the class :
    - isOrdered <- false,
    - isUnique <- false,
    - location is empty,
    - lower <- 1,
    - upper <- 1,
    - type <- Integer (defined in the package Primitive Type),

2.4. ATL Code

```
-- @name    Introducing Primary key
-- @version    1.0
-- @domains    Catalogue of Model Transformations
-- @authors    Baudry Julien (jul.baudry<at>gmail.com)
-- @date    2006/08/02
-- @description   The purpose of this transformation is to introduce a primary key in each
class
-- @see http://www.dcs.kcl.ac.uk/staff/kcl/tcat.pdf
-- @see section 1.6, page 6
-- @see author of article : K. Lano


module PrimaryKey;
create OUT : KM3target from IN : KM3;

--@begin rule Metamodel
rule Metamodel {
   from
      inputMm:KM3!Metamodel
   to
      outputMm:KM3target!Metamodel (
         location <- inputMm.location,
         contents <- inputMm.contents
      )
}
--@end rule Metamodel

--@begin rule Package
rule Package {
   from
      inputPkg:KM3!Package
   to
      outputPkg:KM3target!Package (
         name <- inputPkg.name,
         contents <- inputPkg.contents
      )
}
--@end rule Package

--@begin rule Class
rule Class {
   from
      inputA:KM3!Class
```

```
   to
      outputA:KM3target!Class (
         name <- inputA.name,
         isAbstract <- inputA.isAbstract,
         structuralFeatures <- inputA.structuralFeatures
      ),
      key:KM3target!Attribute (
         name <- inputA.name.toLower()+'Key',
         isOrdered <- false,
         isUnique <- false,
         location <- '',
         lower <- 1,
         upper <- 1,
         type <- KM3!DataType.allInstances()->select(a|a.name = 'Integer')->first(),
         owner <- outputA
      )
}
--@end rule Class

--@begin rule reference
rule DataType {
from
      inputData:KM3!DataType
   to
      outputData:KM3target!DataType(
         name <- inputData.name,
         location <- inputData.location
      )
}
```

3. References

[1] Catalogue of Model Transformations
http://www.dcs.kcl.ac.uk/staff/kcl/tcat.pdf