LABORATORIO 3

ATLAS Transformation Language (ATL)

Objetivo: Ver las características básicas del lenguaje de Transformación ATL.

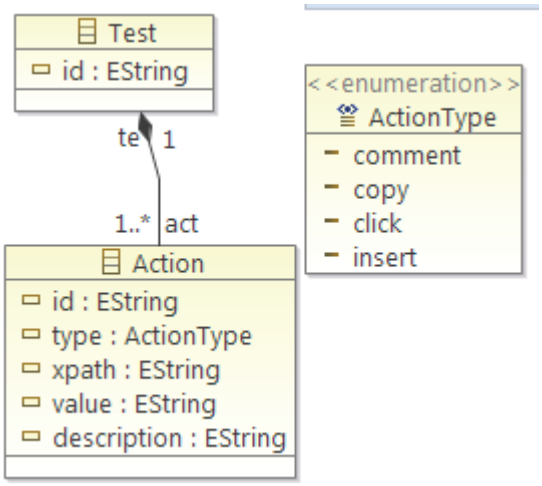
Se proporciona un proyecto de tipo ATL. La carpeta *metamodel* contiene los metamodelos necesarios para los ejercicios. En este laboratorio se tienen que implementar las reglas de transformación, a guardar en la carpeta *rules*. Crear en la carpeta *model* los modelos necesarios para hacer las pruebas.

Para realizar los ejercicios de este laboratorio se utilizarán diferentes variaciones de los siguientes dos metamodelos: *Test* y *TestCase*.

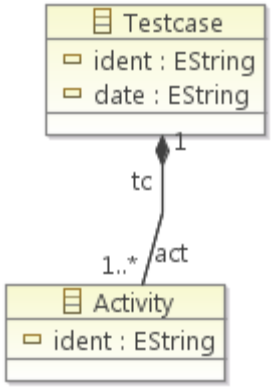
En los siguientes ejercicios se proponen tareas de transformación teniendo en cuenta ciertas partes concretas de los metamodelos. Para implementar los ejercicios se proporcionan los metamodelos, todos ellos basados en un mismo dominio (que aparece completo al final). En cada caso hay que utilizar los que correspondan.

1. Transformación con referencia

Transformar *Test* en *Testcase* y *Action* en *Activity*. No tener en cuenta el atributo *date* que aparece en la clase *Testcase*. Para simplificar, en este caso la clase *Activity* no es abstracta.



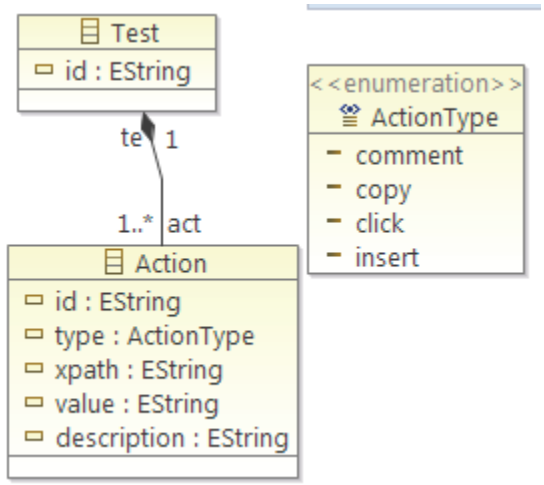
Metamodel: *TestMM-1.ecore*
Model: *Test-1.xmi*



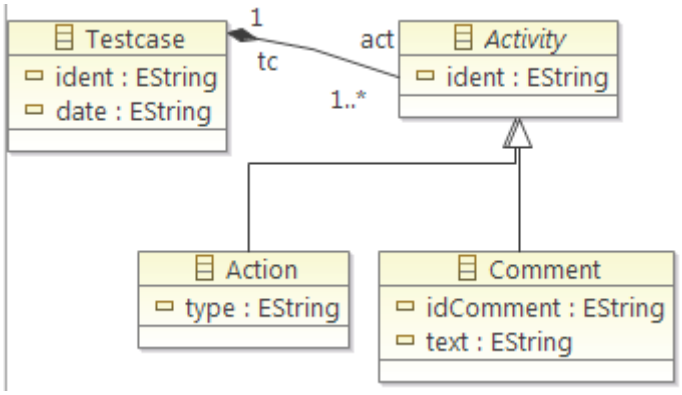
Metamodel: *TestCase-1.ecore*

2. Transformación con especialización

La clase *Activity* es abstracta. La clase *Action* de *Test-1* debe transformarse en *Action* o *Comment* dependiendo del valor de su atributo *type*. El valor de *text* de *Comment* se obtendrá de *description*.



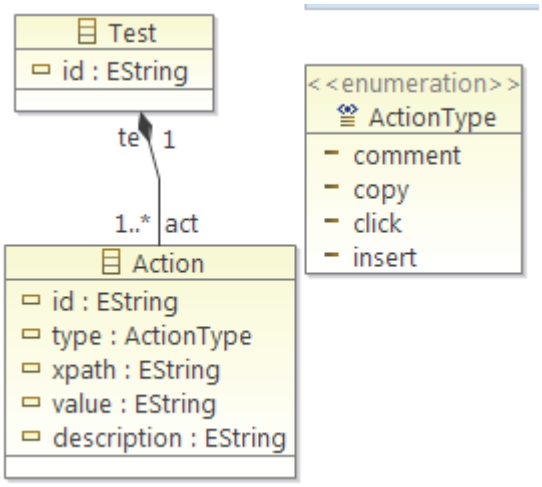
Metamodel: *TestMM-1.ecore*
Model: *Test-1.xmi*



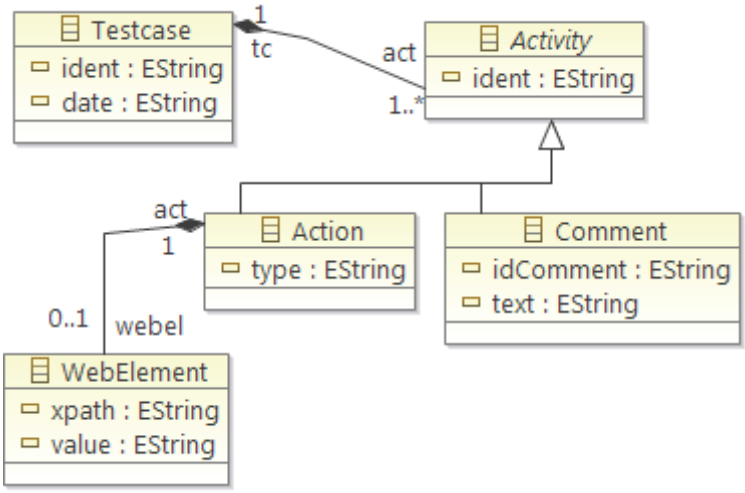
Metamodel: *TestCase-2.ecore*

3. Transformación de 1 a 2 elementos (2 elementos con referencia)

Completar la transformación del ejercicio anterior para generar la clase *WebElement* a partir del contenido de *Action* en *TestMM-1*.



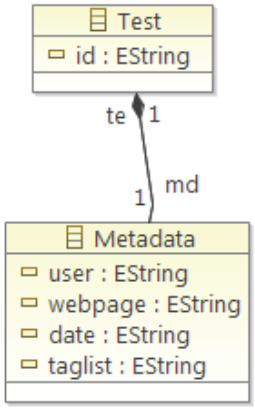
Metamodel: *TestMM-1.ecore*
Model: *Test-1.xmi*



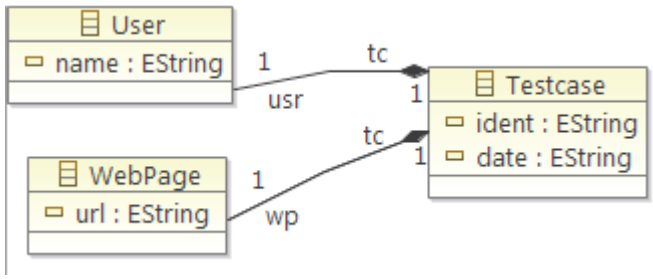
Metamodel: *TestCase-3.ecore*

4. Transformación de 1 a 2 elementos (2 elementos sin referencia)

Transformar *Test* en *Testcase* y *Metadata* en *User* y *WebPage*.



Metamodel: *TestMM-2.ecore*
Model: *Test-2.xmi*



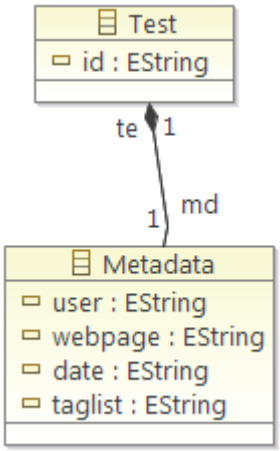
Metamodel: *TestCase-4.ecore*

5. Transformación de 1 a 3 elementos (3 elementos sin referencia)

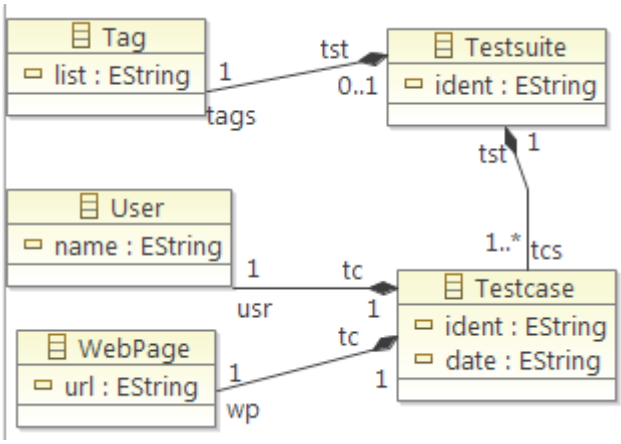
a) Transformar *Test* en *Testcase* y *Testsuite*, y además, *Metadata* en *User*, *WebPage* y *Tag*. Se deben definir 2 reglas.

b) Transformar *Test* en *Testcase*, y *Metadata* en *User*, *WebPage* y *Tag*. Para generar *Testsuite* usar una regla de tipo *endpoint*.

c) Transformar *Test* en *Testcase*, y *Metadata* en *User*, *WebPage* y *Tag*. Para generar *Testsuite* usar una regla de tipo *entrypoint*. Pista: Crea un *helper* de tipo atributo para el contexto de *Testsuite*. La regla *entrypoint* debe crear la instancia *Testsuite* y asignar su referencia al *helper* anterior. Después, a la hora de crear las reglas de *Tag* y *Testcase*, en lugar de rellenar las referencias *tags* y *tcs*, utiliza las opuestas (i.e. *tst* y *tst*).



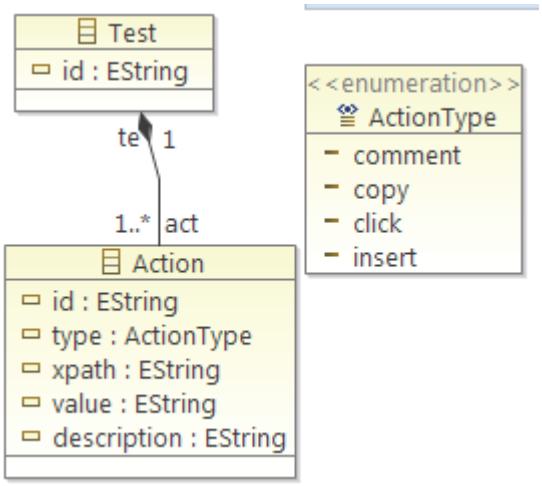
Metamodel: *TestMM-2.ecore*
Model: *Test-2.xmi*



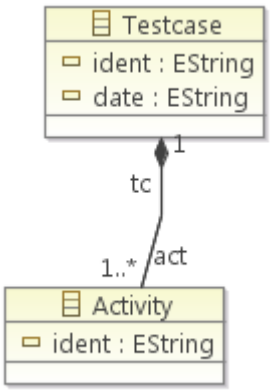
Metamodel: *TestSuite.ecore*

6. Filtros en los *bindings*

Transformar *Test* en *Testcase* y los *Action* que sean de tipo *comment* en *Activity* (los que no sean *comment* no se tienen en cuenta). Para simplificar, en este caso la clase *Activity* no es abstracta. Atención con el *binding* de la característica *act* de *Activity*.



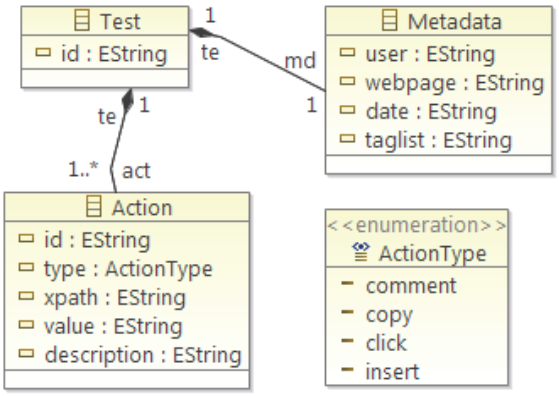
Metamodel: *TestMM-1.ecore*
Model: *Test-1.xmi*



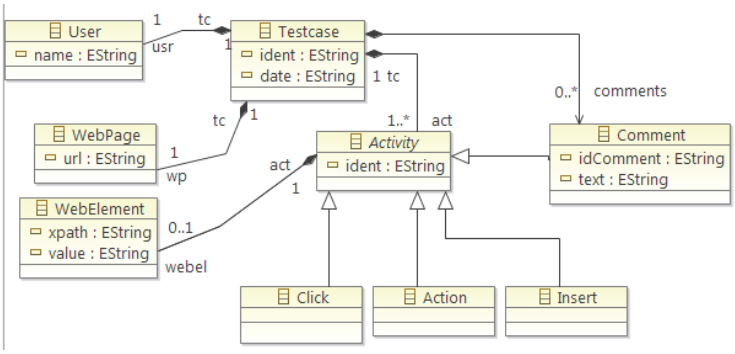
Metamodel: *TestCase-1.ecore*

7. Definir *helpers* con contexto

Definir un *helper* de nombre *getDate* para que calcule el valor para el atributo *date* de la clase *Testcase*, a partir del valor de *date* en la clase *Metadata*. Definir otro *helper*, de nombre *realActions*, para que dado un *Test*, devuelva una colección con las acciones que no son comentario (*Comment*). Implementar la transformación entre los metamodelos *TestMM* y *TestCase* basándose en las reglas de los ejercicios anteriores (Nótese que las acciones se convierten en la acción correspondiente en lugar de todas juntas).



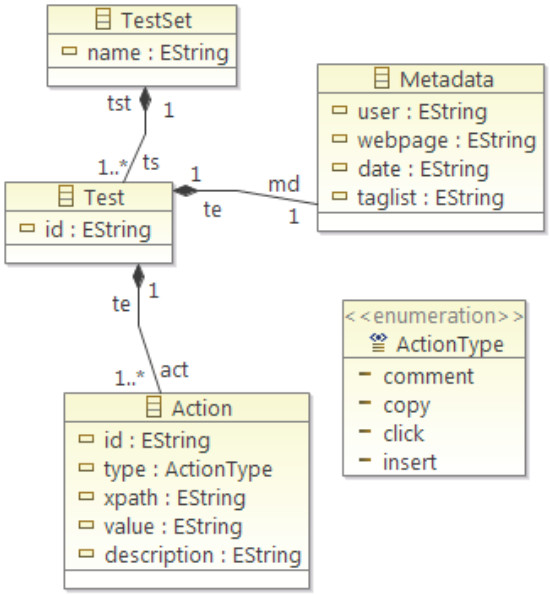
Metamodel: T*estMM.ecore*
Model: *Test-2.xmi*



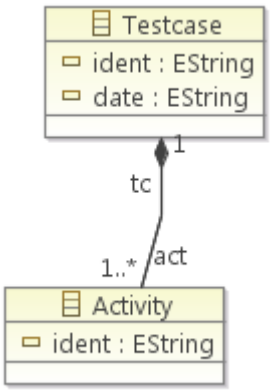
Metamodel: *TestCase.ecore*

8. Definir *helper* sin contexto

Transformar *TestSet* en *Testcase* y *Action* en *Activity*. El valor de *ident* en *Testcase* se calculará así: la suma de los identificadores de las acciones (*Action*) relacionados con los elementos *Test* que ha creado el usuario *'admin'*. Para simplificar, en este caso la clase *Activity* no es abstracta.



Metamodel: *TestMM-5.ecore*
Model: *TestSet-1.xmi*



Metamodel: *TestCase-1.ecore*

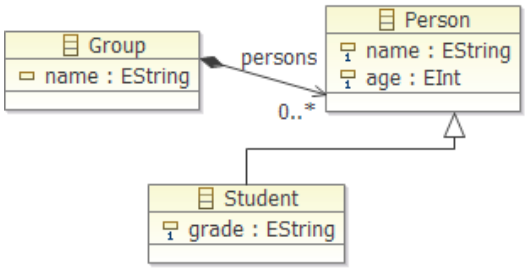
9. Transformación con especialización y relación distribuida

Transformar los modelos conformes al metamodelo *PersonsOne.ecore* en modelos conformes al metamodelo *PersonsTwo.ecore*. La clase *Person* se transformará en *Worker* o *Retired* (dependiendo del valor de *age*, y con el valor 65 como límite), y la clase *Student* en *Undergraduate*.

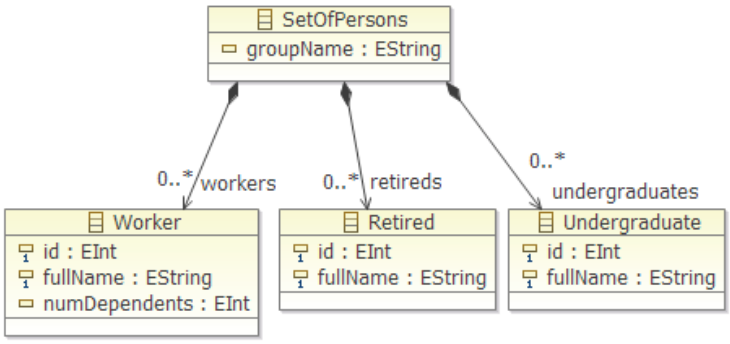
a) En un primer momento, no tener en cuenta los atributos *id* ni *numDependents* que aparecen en las metaclases del metamodelo *PersonsTwo.ecore*.

b) Modificar la transformación del apartado anterior para que se asigne un identificador único a cada elemento creado (*Worker, Retired* y *Undergraduate*). ¿Quizás utilizando un helper de atributo y sentencias imperativas?

c) Modificar la transformación del apartado anterior para que se calcule el número dependientes que tiene una persona. Para las personas mayores de 65 años y para los estudiantes no hay que hacer nada nuevo, pero para los trabajadores menores de 65 en el valor de *numDependents* hay que asignar el número de personas que dependen de ellos. Los dependientes de un trabajador son los estudiantes y retirados que tienen su mismo nombre.



Metamodel: *PersonsOne.ecore*
Model: *personsinput.xmi*



Metamodel: *PersonsTwo.ecore*