Procesadores de Lenguajes 2

Transformaciones de modelo a texto con MOFScript

Curso 2013-2014

Iván Ruiz Rube
Departamento de Ingeniería Informática
Escuela Superior de Ingeniería
Universidad de Cádiz

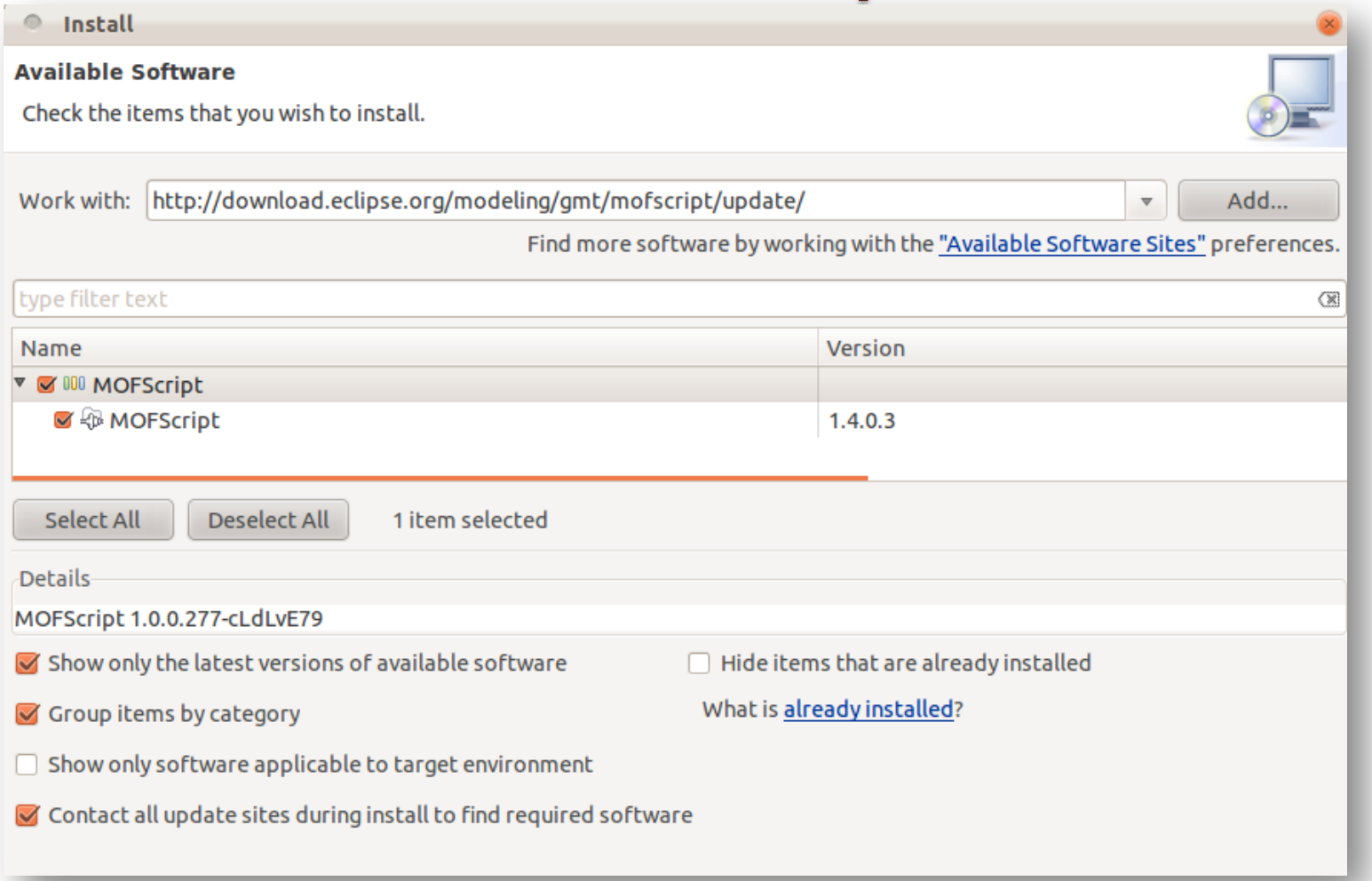
Contenidos

- Instalación
- Introducción
- Componentes
- Características del lenguaje
- Desarrollo de una transformación

TRANSFORMACIONES DE MODELO A TEXTO CON MOFSCRIPT

INSTALACIÓN

Instalación MOFScript



`Help → Install New Software`

TRANSFORMACIONES DE MODELO A TEXTO CON MOFSCRIPT

INTRODUCCIÓN

MOFScript

- Herramienta muy utilizada para transformaciones de modelo a texto (código o documentación).
- Implementación de la especificación M2T de la OMG. Alineamiento con el estándar QVT.
- Existen alternativas M2T más recientes como Acceleo, JET o Xtend.

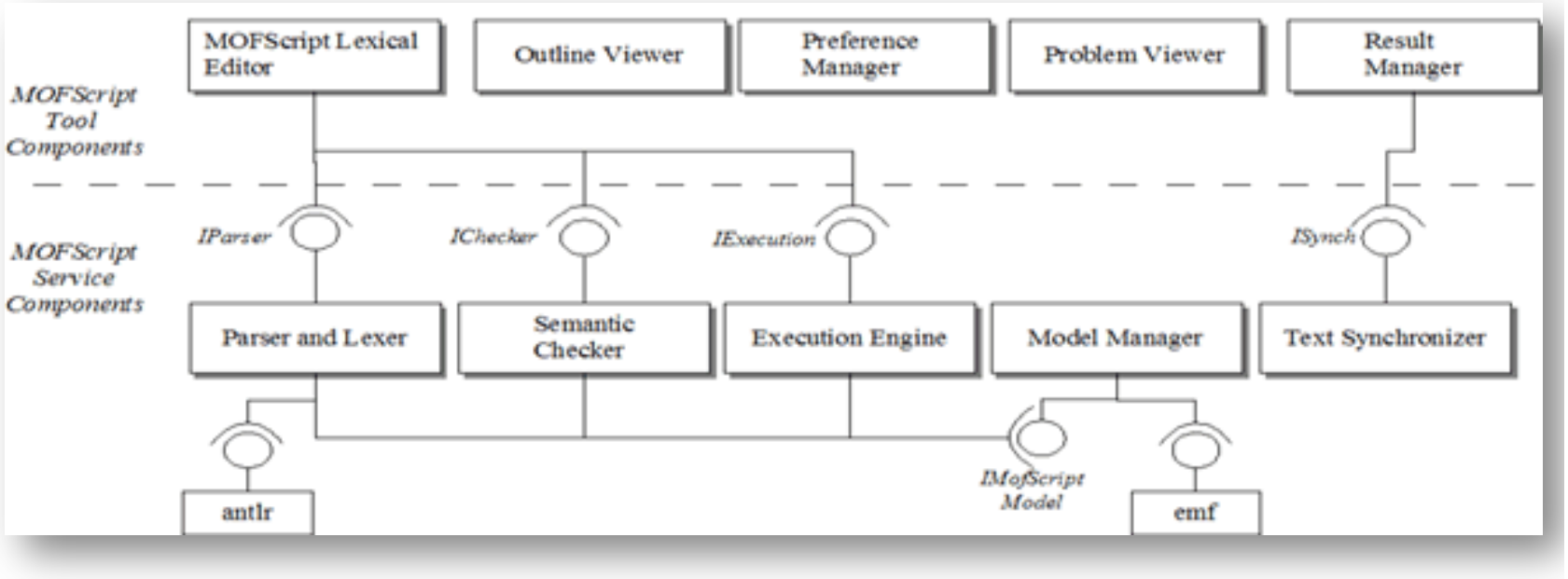
Características (I)

- Generación de texto desde modelos basados en MOF, UML, Ecore u otros creados con EMF.
- Permite crear múltiples ficheros de salida. Codificación configurable.
- Control de la trazabilidad desde el modelo a los bloques de código generado.
- Integración con código Java externo.

Características (II)

- Fácil de usar: lenguaje sencillo con pocos constructores.
- Lenguaje imperativo, similar a los lenguajes de scripting.
- Reglas invocadas explícitamente, excepto la inicial (main).
- Se permite la herencia de transformaciones.

Arquitectura



MOFScript se distribuye como un plugin que incorpora las herramientas necesarias para que el usuario pueda interaccionar con los servicios de reconocimiento y verificación de los programas, ejecución de las transformaciones, etc.

TRANSFORMACIONES DE MODELO A TEXTO CON MOFSCRIPT

COMPONENTES

Componentes

- Los módulos MOFScript se componen de los siguientes elementos:
  - Declaración del módulo
  - Reglas de transformación
    - Incluyendo una regla *main*.
  - Operaciones de escritura en ficheros

Declaración del módulo

- Define el nombre del módulo de transformación, el/los metamodelo(s) de origen y opcionalmente, la importación de otros módulos.

```
import "otraTransformacion.m2t"
texttransformation  module_name
    (in InputMM1:"input_metamodel_uri1",
    in InputMM2:"input_metamodel_uri2")
{
...
```

Reglas de transformación

- Las reglas de transformación son similares a los métodos en Java. Pueden tener un contexto, un tipo de retorno, parámetros y un conjunto de sentencias a ejecutar. Debe existir una regla *main*, sin parámetros y sin tipo de retorno.

```
[InputMM.MMElement::] rule_name (param1:Type1, param2:Type2) {
    // statements
    ...
    result= exp;
}
```

Escritura en ficheros

- Indicar el nombre (relativo o absoluto) del fichero donde se escribirá el texto de salida. Las sentencias *println* y el texto '*escapado*' permitirán escribir en el último *file* declarado.

```
file fichero("ficherodesalida.txt");
fichero.println("escribiendo texto...");
nl(1); tab(2); space(3);
` escribiendo más texto...';
```

TRANSFORMACIONES DE MODELO A TEXTO CON MOFSCRIPT

CARACTERÍSTICAS DEL LENGUAJE

Expresiones MOFScript

```
property name:Type = value;
var name:Type = value;
```

```
if (condition) {
    // statements
} else  {
    // statements
}
```

```
while (condition){
    //statements
}
```

Tipos de datos MOFScript

- Predefinidos
  - String
  - Integer
  - Real
  - Boolean
  - List
  - Hashtable
  - PropertyMap
  - Object
- Tipos del metamodelo (metaclases)

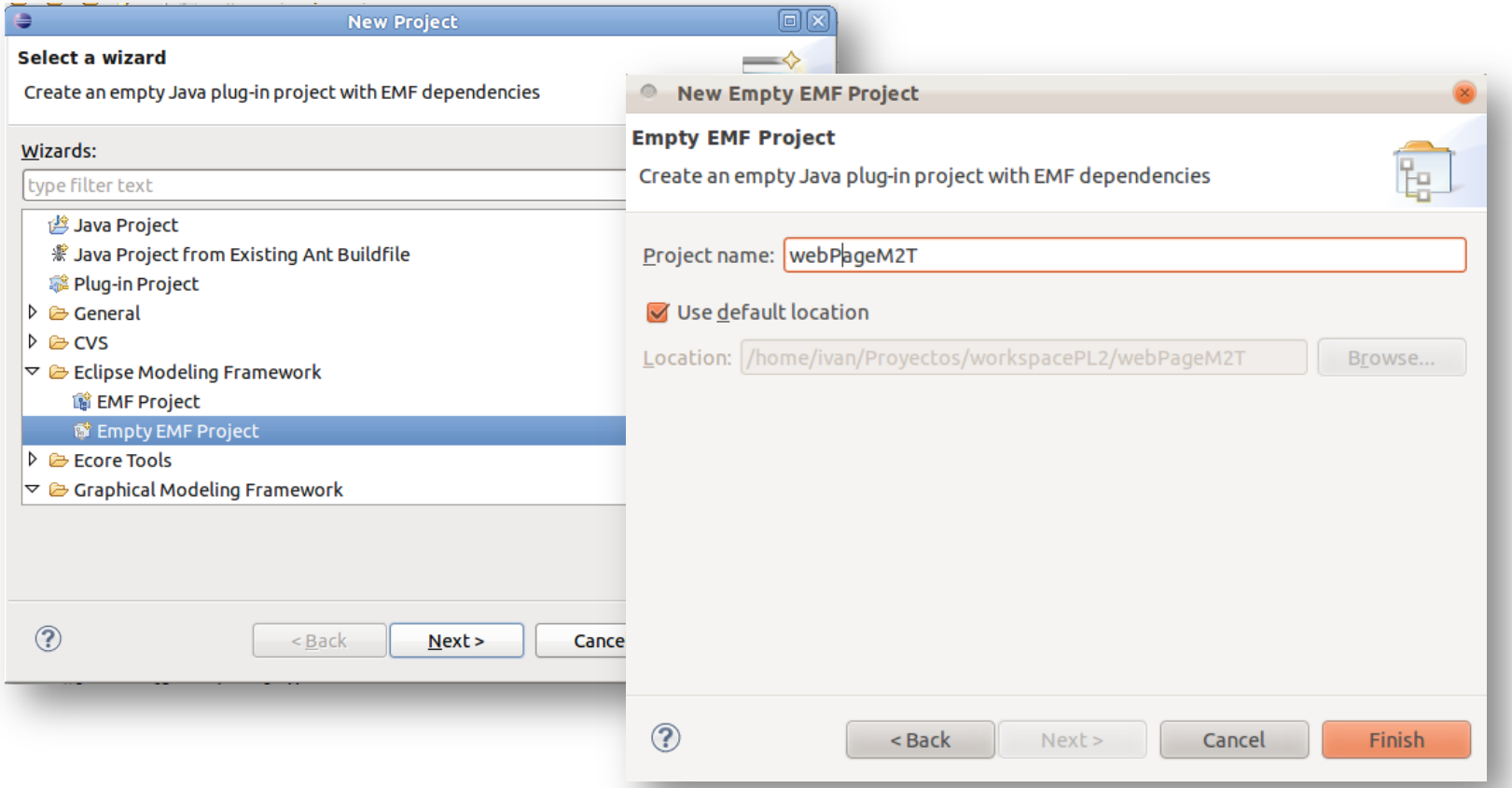
Operaciones MOFScript

- Predefinidos:
  - String: *substring()*, *toLower()*, *replace()*
  - Integer: +, -, *, /
  - List: *add()*, *remove()*, *forEach()*, *select()*
  - Hashtable: *put()*, *get()*, *keys()*
  - PropertyMap: *load()*, *loadXML()*, *storeXML()*
- Tipos del metamodelo
  - *objectsOfType()*, *oclIsTypeOf()*, *oclIsKindOf()*
- Utilidades del sistema
  - *position()*, *time()*, *date()*,

TRANSFORMACIONES DE MODELO A TEXTO CON MOFSCRIPT

DESARROLLO DE UNA TRANSFORMACIÓN

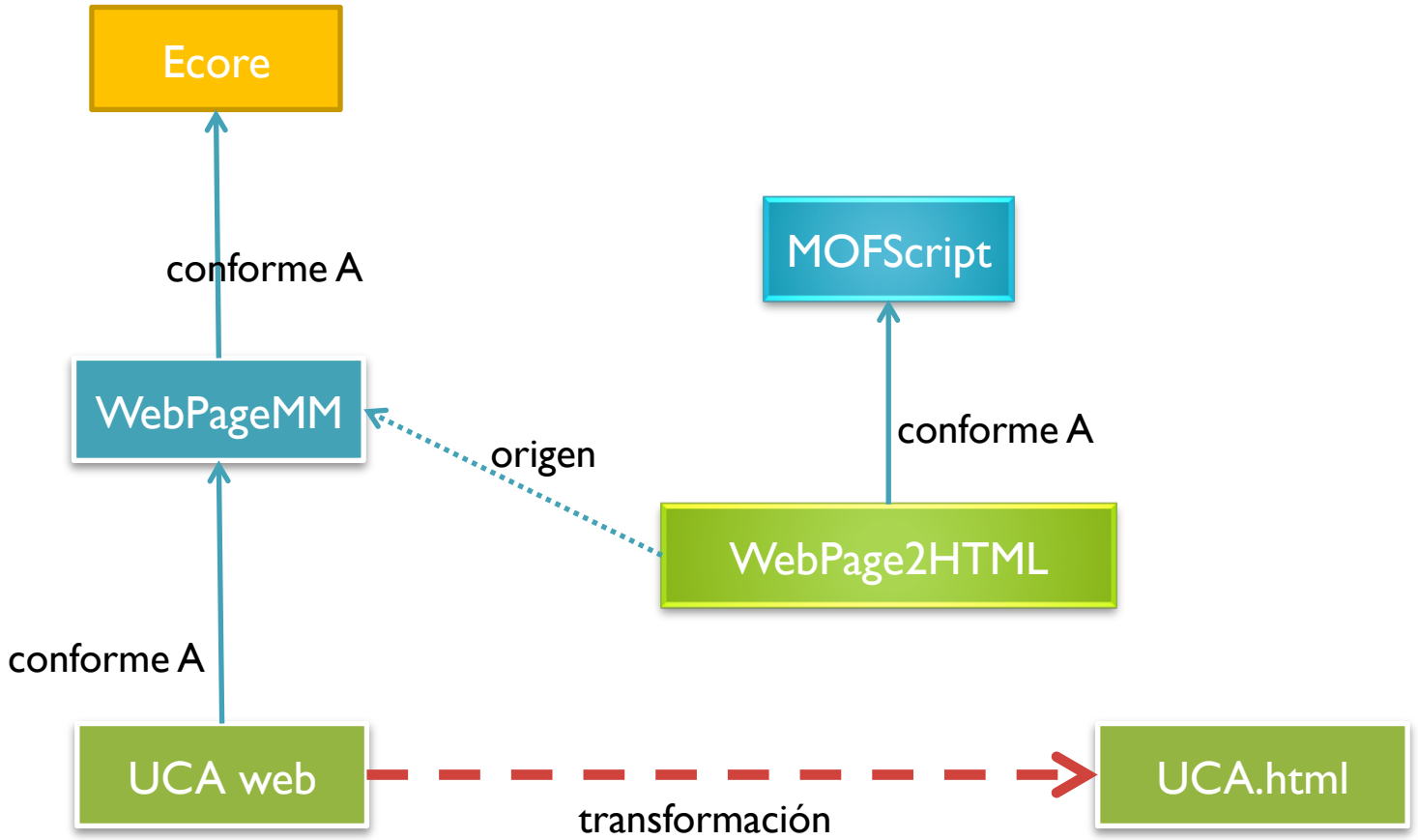
Creación de un proyecto



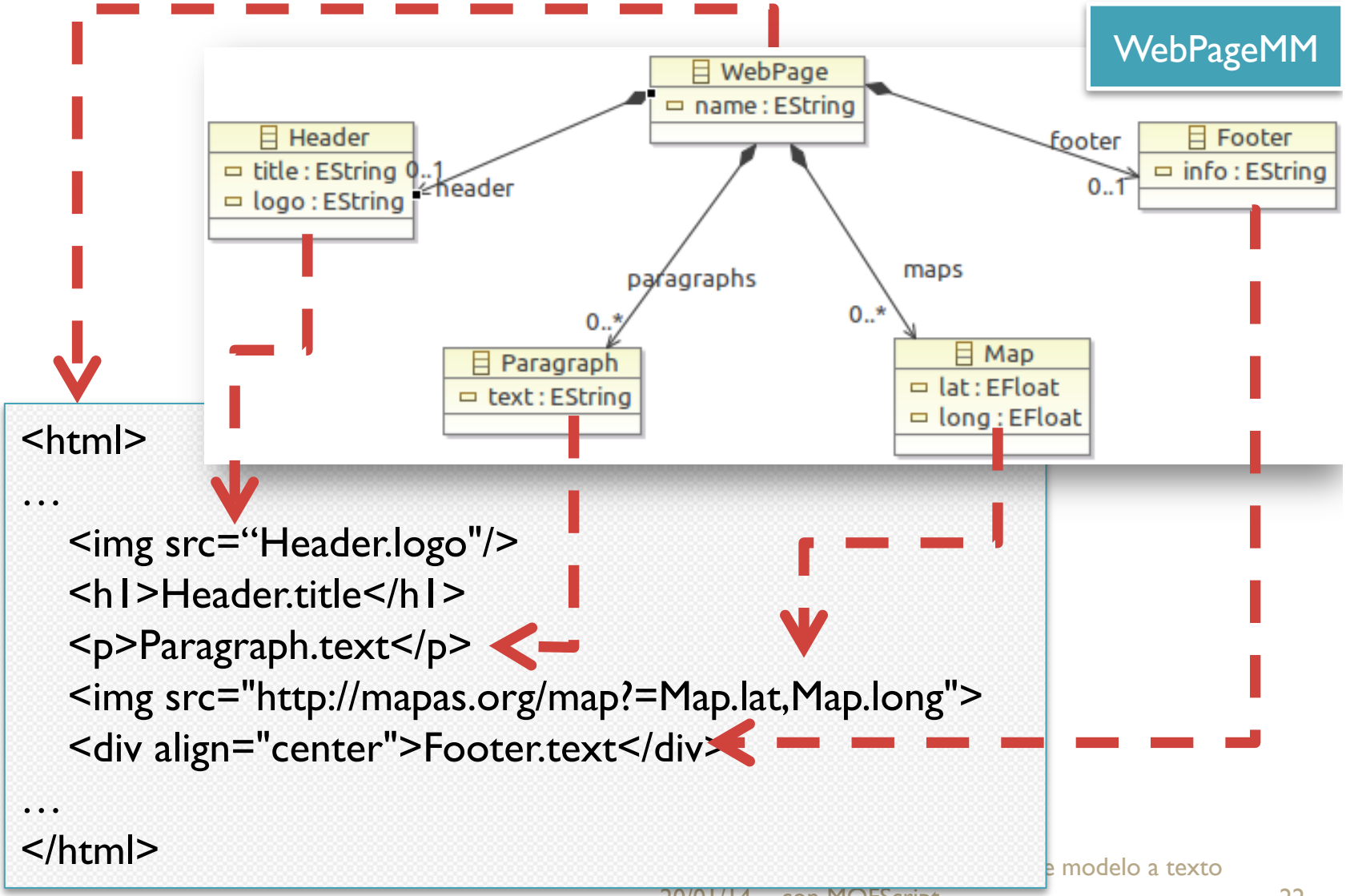
`File → New Project → Empty EMF Project`

Los metamodelos los crearemos dentro de un proyecto EMF

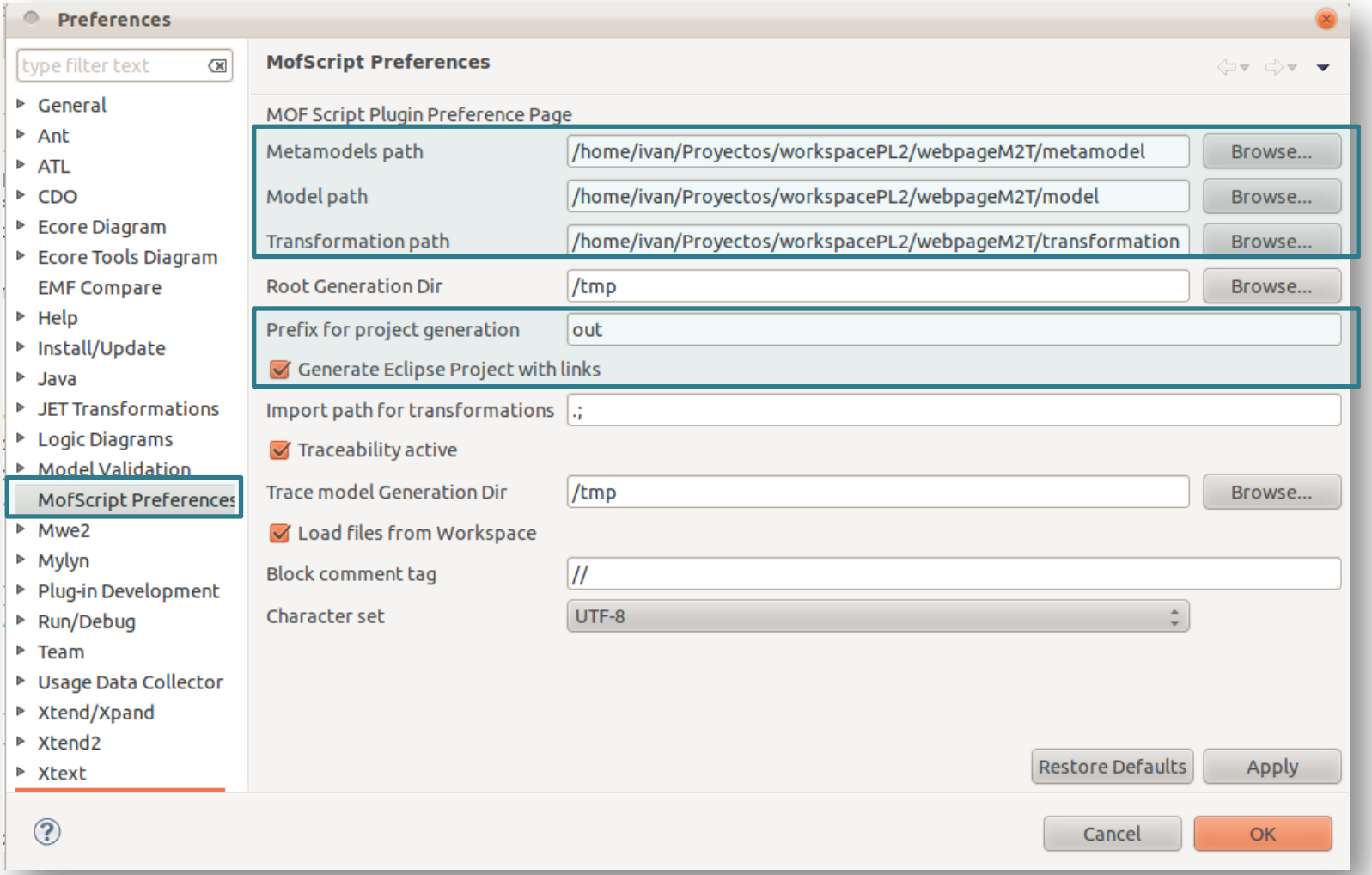
Arquitectura del ejemplo



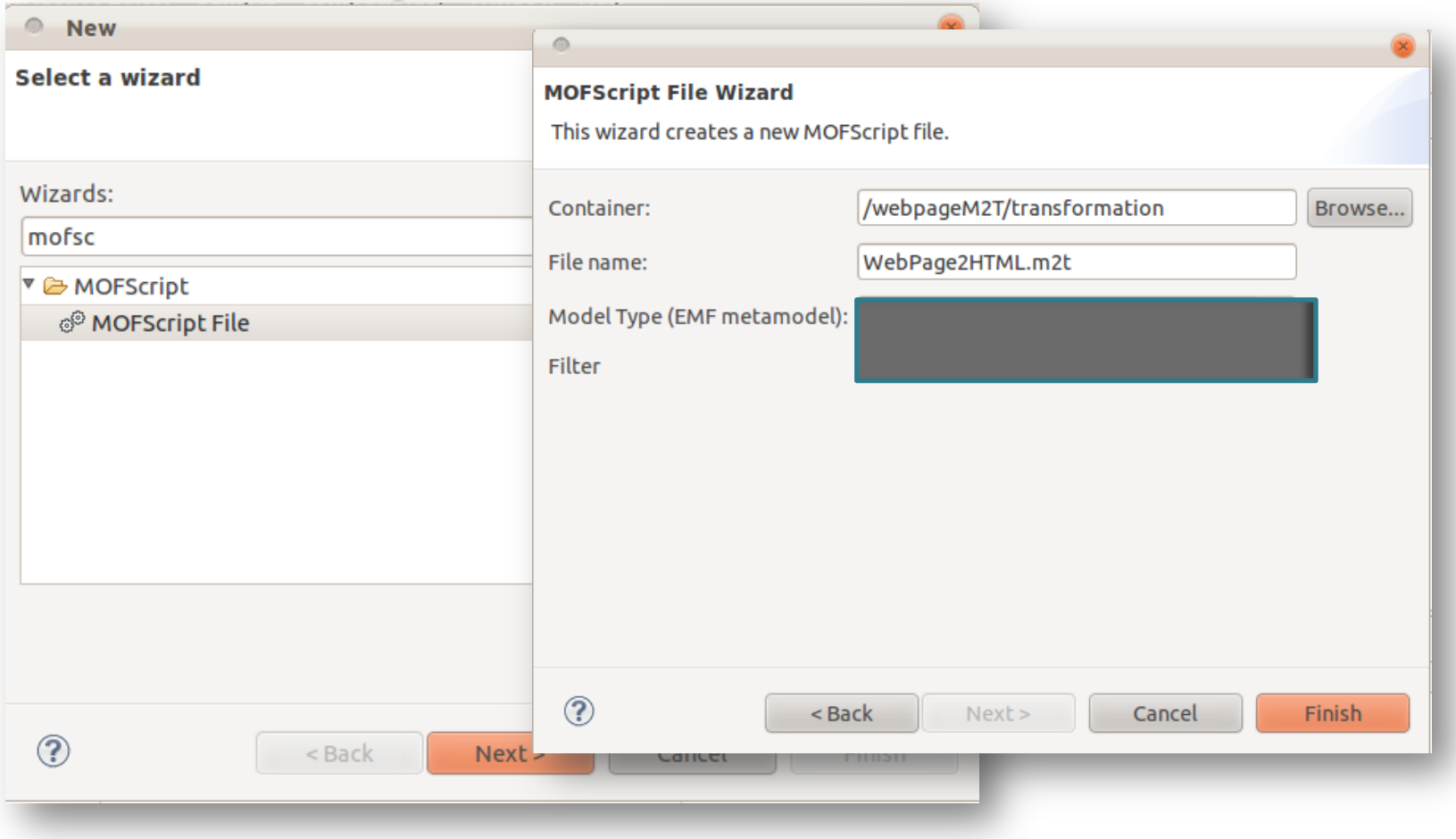
Correspondencia del metamodelo



Establecer rutas de trabajo



Crear una transformación MOFScript



`File → New → MOFScript File`

Desarrollo de la transformación (I)

- Nuestro fichero M2T definirá un módulo de transformación, el cual transformará un modelo basado en el metamodelo de páginas web en texto.

```
property filename:String = "index.html";

texttransformation WebPage2HTML (in WebPageMM:"http://
    webpagemm/1.0") {
```

Desarrollo de la transformación (II)

- Función principal, la cual abrirá el fichero de salida, escribirá las etiquetas de apertura del HTML, lo poblará y luego escribirá las etiquetas de cierre.

```
WebPageMM.WebPage::main () {
  stdout.println ("Generando web para... " + self.name);
  file (filename); // Creamos archivo
  writeHTMLHeader(self.name); // Apertura del fichero HTML
  self.toHTML(); // Contenidos del HTML
  writeHTMLEnd(); // Cierre del fichero HTML
 }
```

Desarrollo de la transformación (III)

- Función auxiliar que genera las etiquetas de cabecera del documento HTML.

```
writeHTMLHeader(name:String){
  '<html lang="es">\n';
  '<head>\n';
  tab(1);  '<meta http-equiv="content-type" content="text/html;
    charset=utf-8">\n';
  tab(1);  '<title>'; name;  '</title>\n';
  '</head>\n';
  '<body>\n';
 }
```

Desarrollo de la transformación (IV)

- Función auxiliar que genera las etiquetas de cierre del documento HTML.

```
writeHTMLEnd(){
  nl(1);
  '</body>';
  nl(1);
  '</html>';
 }
```

Desarrollo de la transformación (V)

- Función auxiliar que genera el contenido HTML desde el modelo origen.

```
WebPageMM.WebPage::toHTML() {
  self.header.toHTML();
  self.paragraphs->forEach(paragraph:WebPageMM.Paragraph){
        paragraph.toHTML();
  }
  self.maps->forEach(map:WebPageMM.Map){
        map.toHTML();
  }
  self.footer.toHTML();
}
```

Desarrollo de la transformación (VI)

- Función auxiliar que genera el HTML necesario para representar el título y el logotipo de la página web.

```
WebPageMM.Header::toHTML() {
  nl(1);tab(1);
  '<img width="300" height="126" src="' self.logo '"/>';
  nl(1);tab(1);
  '<H1>' self.title '</H1>';
}
```

Desarrollo de la transformación (VII)

- Función auxiliar que genera el HTML necesario para representar un párrafo de la página web.

```
WebPageMM.Paragraph::toHTML() {
    nl(1);tab(1);
    '<p>' self.text '</p>';
}
```

Desarrollo de la transformación (VIII)

- Función auxiliar que genera el HTML necesario para representar un mapa estático de Google.

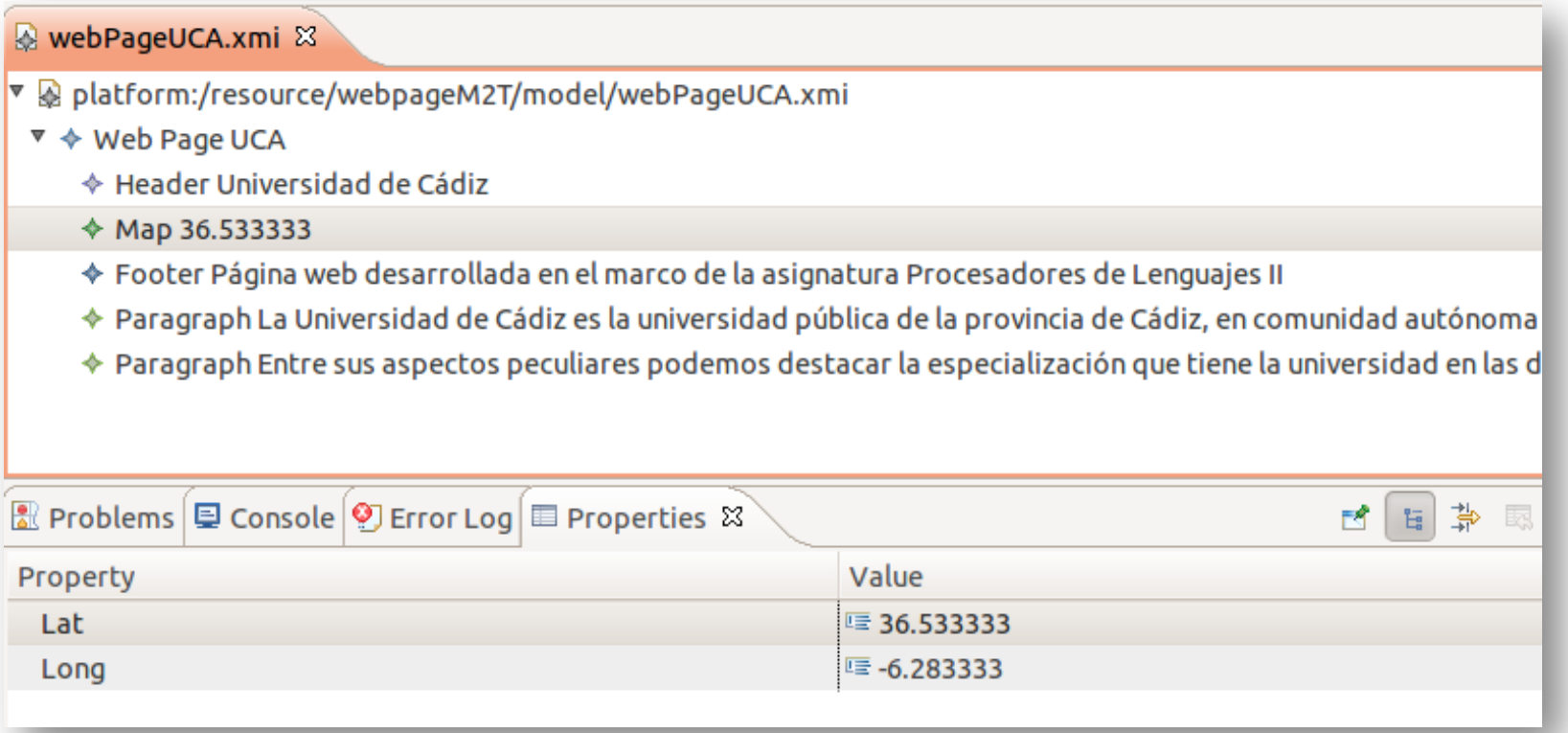
```
WebPageMM.Map::toHTML() {
    nl(1); '<br/>';
     nl(1); tab(1);
     '<img src= "http://maps.googleapis.com/maps/api/staticmap?
    center=' self.lat ',' self.long ' &zoom=11& size=200x200
    &sensor=false">';
 }
```

Desarrollo de la transformación (IX)

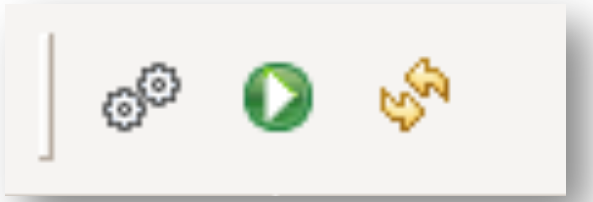
- Función auxiliar que genera el HTML necesario para representar la información del pié de página.

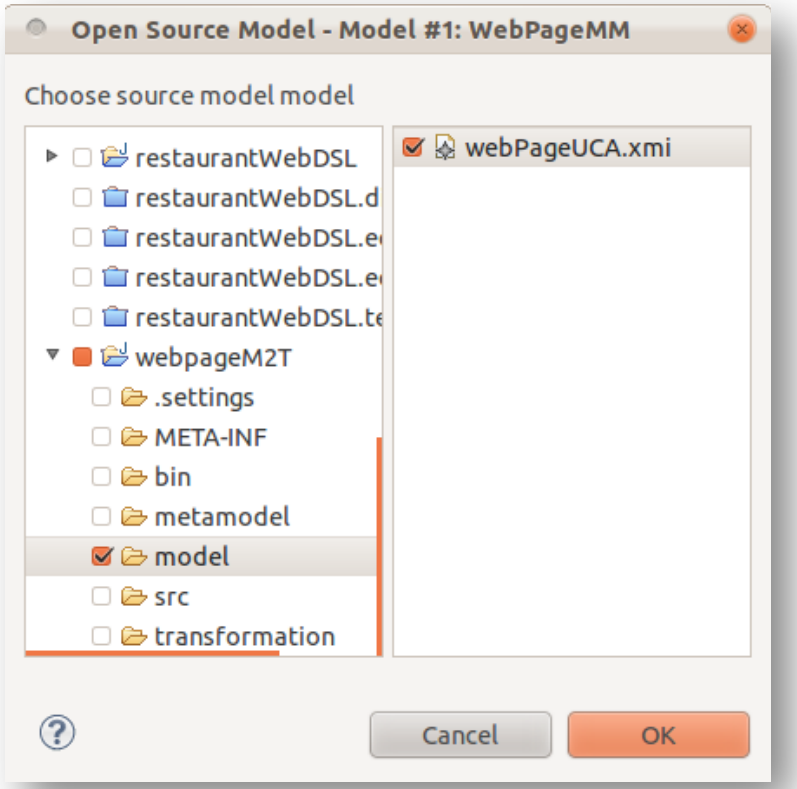
```
WebPageMM.Footer::toHTML() {
    nl(1);
    '<br/><hr/>';
    nl(1);tab(1);
   '<div align="center">' self.info '</div>';
 }
```

Desarrollo de un modelo origen



Ejecución de la transformación





Las opciones MOFScript de la barra de botones de Eclipse, permiten compilar el fichero M2T, ejecutar una transformación para un modelo dado y ejecutar la transformación anterior.

Resultado: Página HTML

TRANSFORMACIONES DE MODELO A TEXTO CON MOFSCRIPT

RESUMEN

¿Qué hemos aprendido hoy?

- MOFScript es un lenguaje y un entorno de ejecución para generar texto a partir de modelos (M2T).
- Permite generar varios ficheros de salida de código ejecutable o documentación.
- El desarrollo de las reglas de transformación es similar a la programación de métodos en Java.

Procesadores de Lenguajes 2

Transformaciones de modelo a texto con MOFScript

Iván Ruiz Rube
ivan.ruiz@uca.es