ECLIPSE MODELING FRAMEWORK

Desarrollo Automatizado de Software

4º Ingeniería Informática

Universidad Autónoma de Madrid

Eclipse Modeling Framework (EMF)

- Framework de modelado sobre Eclipse

- A partir de una especificación del modelo de datos en XMI, produce las clases de implementación Java para el modelo, y un editor básico para editar instancias del modelo.

- Página web: http://www.eclipse.org/modeling/emf
- Descarga (Eclipse Modeling Tools)
  - Eclipse Modeling Tools: http://www.eclipse.org/downloads
  - Update site: http://download.eclipse.org/modeling/emf/emf/updates/releases/
- Documentación: http://www.eclipse.org/modeling/emf/docs/
- Tutorial: http://www.vogella.de/articles/EclipseEMF/article.html

Ficheros .ecore y .genmodel

- **ecore**: definición del meta-modelo (o *modelo de dominio*). Un ecore tiene un elemento raíz que representa el meta-modelo, con paquetes que contienen la definición de:
  - EClass: clase, puede tener 1 ó más atributos y referencias
  - EAttribute: atributo, tiene un nombre y un tipo
  - EReference: relación que va de una clase a otra
  - EDataType: tipo de un atributo

- **genmodel**: información para la generación de código (ej. path)

Generación de código Java

- Generación de código a partir de .ecore y .genmodel:
  - `model`: interfaces y factoría para la creación de objetos
  - `model.impl`: implementación de las interfaces
  - `model.util`: adapter factory

- A partir de cada clase se genera una interfaz con getter/setters. Cada setter notifica a los observadores del modelo.

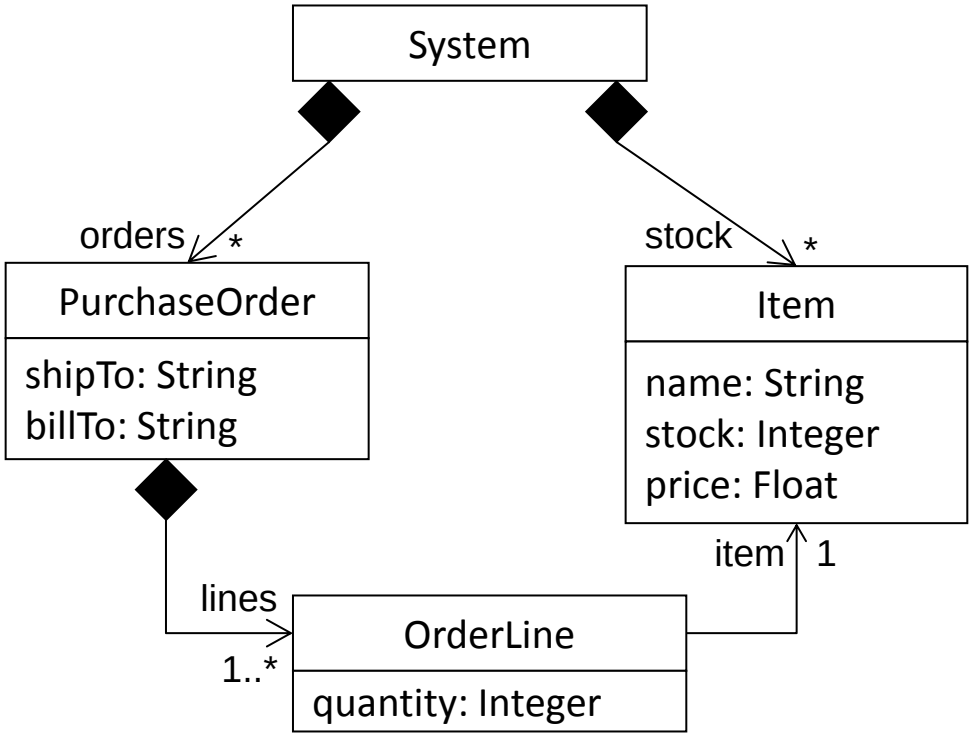
- Los métodos generados están anotados con `@generated`. Si se regenera el código, los métodos anotados con `@generated` se sobreescriben.

Generación de editor

- A partir del.ecore y .genmodel, también se puede generar un editor (en forma de árbol) para las instancias del meta-modelo.

- El editor es un plugin de Eclipse.

Ejemplo



Lenguaje para modelar el stock de productos de una empresa, y los pedidos que sirve a sus clientes.

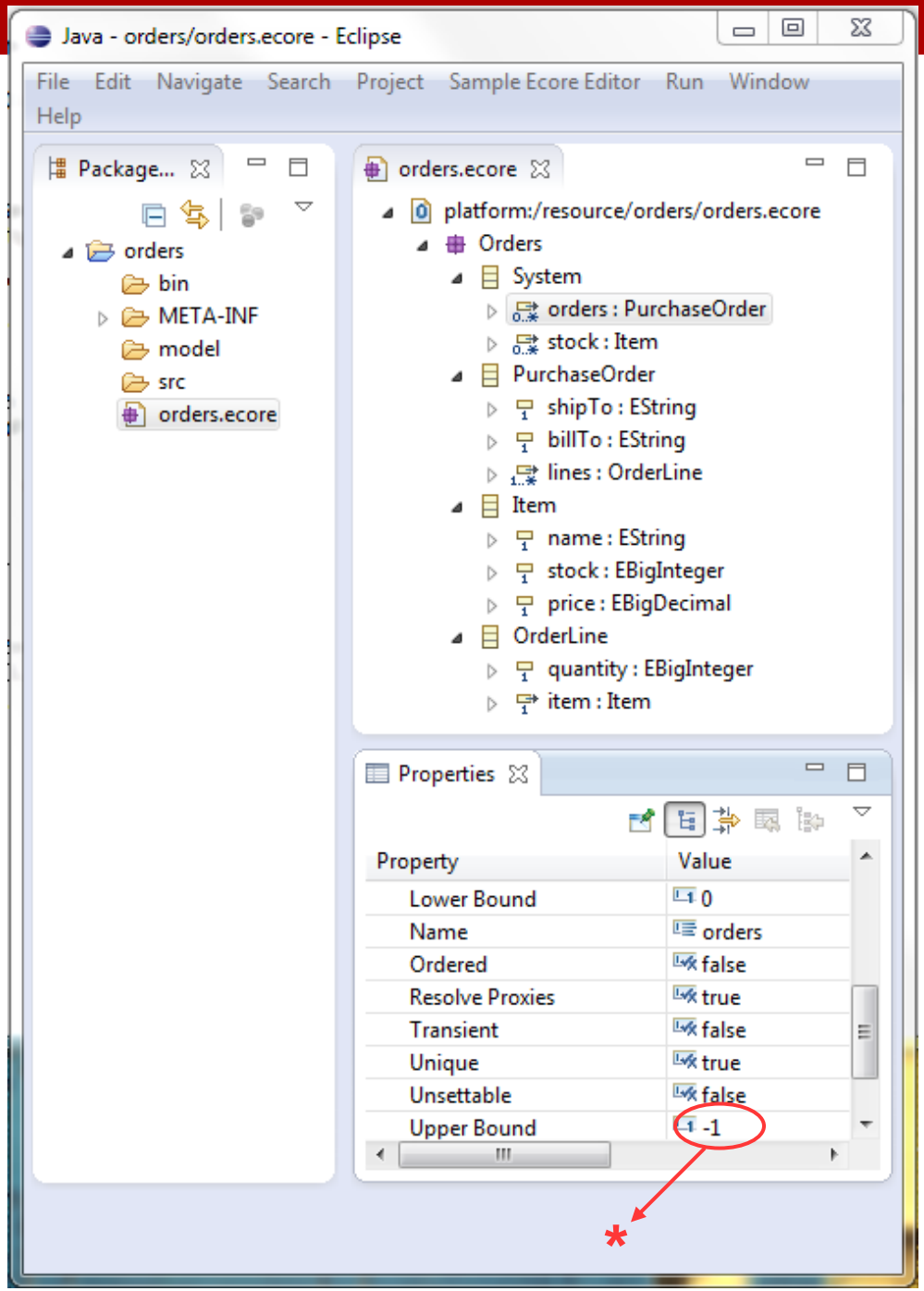
En *moodle* tienes el proyecto EMF para este meta-modelo.

Eclipse Modeling Framework
Cómo definir un meta-modelo y generar código

1. **Crear fichero .ecore (meta-modelo)**
   - Crear nuevo proyecto EMF vacío
     File / New / Other / Eclipse Modeling Framework / Empty EMF Project
   - Crear modelo .ecore
     File / New / Other / Eclipse Modeling Framework / Ecore model
   - Añadir clases, atributos y referencias
2. Crear modelo generador (.genmodel)
3. Generar código Java
4. Generar editor de modelos en forma de árbol

Ecore
Elementos

- **EPackage:** un paquete por defecto
  - *Ns Prefix*
  - *Ns URI*
- **EClass**: clases
  - Define atributos y referencias
  - *ESuperTypes*: superclases
  - *Abstract*: case abstracta
- **EAttribute**: atributos
  - *EType*: tipo (int, float, ...)
  - *Lower Bound:* cardinalidad mínima
  - *Upper Bound*: cardinalidad máxima
- **EReference**: fin de asociación
  - *EType*: clase referenciada
  - *Containment*: relación de contenido
  - *Lower Bound:* cardinalidad mínima
  - *Upper Bound*: cardinalidad máxima
  - *EOpposite*: fin de asociación opuesto (para asociaciones bidireccionales)



Ecore

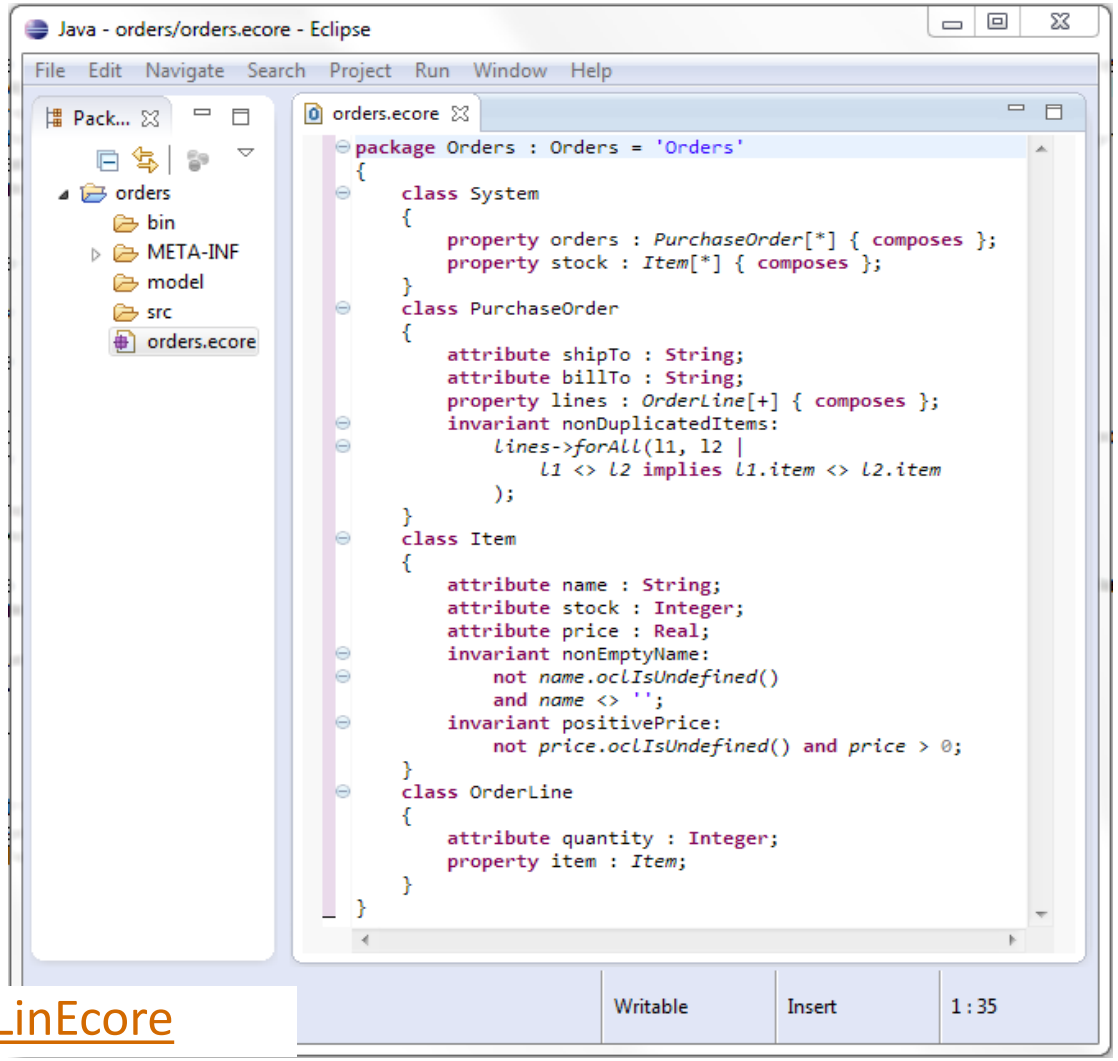
Editor textual de ficheros ecore: OCLinEcore

Instalación:

- *Help / Install new software*
- En el campo de texto *Work with,* seleccionar la entrada *Kepler - http://download.eclipse.org/releases/kepler*
- En la categoría *Modeling*, seleccionar *OCL Example and Editors*

Editor textual: *open with…*
OclInEcore (Ecore) editor

Editor árbol: *open with…*
Sample Ecore Model Editor



http://wiki.eclipse.org/MDT/OCLinEcore

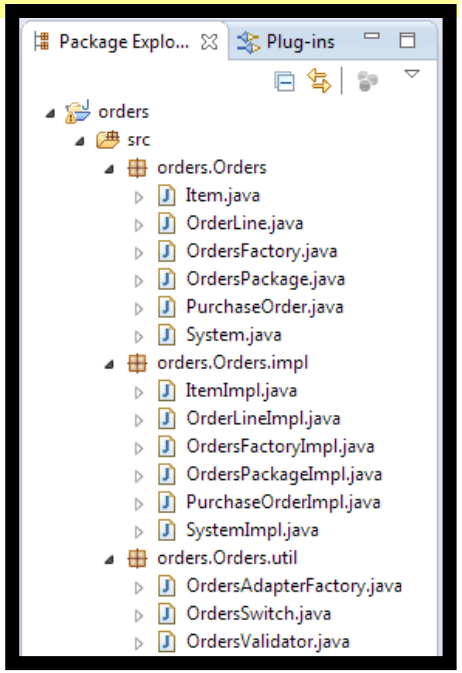
Eclipse Modeling Framework
Cómo definir un meta-modelo y generar código

1. Crear fichero .ecore (meta-modelo)
2. **Crear modelo generador (.genmodel)**
   - Seleccionar fichero .ecore, botón derecho del ratón
     New / Other / Eclipse Modeling Framework / EMF Generator Model
   - Abrir fichero .genmodel generado
   - Seleccionar paquete y editar *All / Base Package*
   - Si el fichero .ecore cambia, *reload...* modelo generador
3. Generar código Java
4. Generar editor de modelos en forma de árbol

Eclipse Modeling Framework

Cómo definir un meta-modelo y generar código

1. Crear fichero .ecore (meta-modelo)
2. Crear modelo generador (.genmodel)
3. **Generar código Java**
   - Abrir modelo generador, botón derecho en raíz
     Generate Model Code
4. Generar editor de modelos en forma



Clases Java generadas
Factoría e interfaces

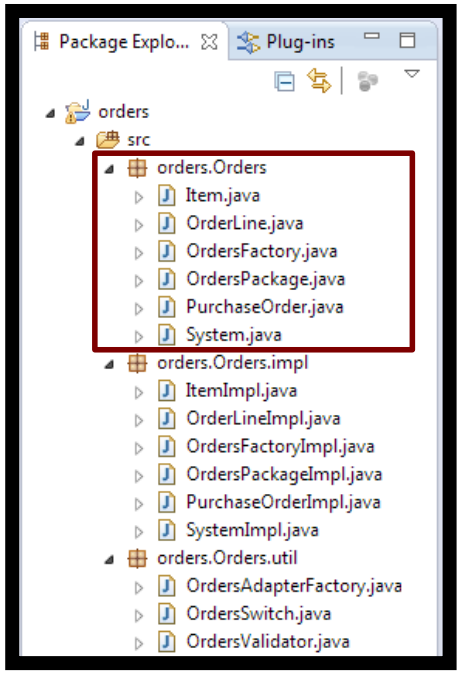
Se genera una clase factoría:

```
public interface OrdersFactory extends EFactory {
  OrdersFactory eINSTANCE = orders.Orders.impl.OrdersFactoryImpl.init();

  System        createSystem();
  PurchaseOrder createPurchaseOrder();
  Item          createItem();
  OrderLine     createOrderLine();
  OrdersPackage getOrdersPackage();
}
```

Se genera una interfaz y una clase de implementación por cada clase del .ecore:

```
public interface Item extends EObject {
  String getName();
  void   setName(String value);

  BigInteger getStock();
  void       setStock(BigInteger value);

  BigDecimal getPrice();
  void       setPrice(BigDecimal value);
}
```

- capacidades reflexivas (ej. `eClass()` devuelve la clase del meta-modelo, `eGet()`, `eSet()`, etc.)
- `EObject` extiende `Notifier` (patrón *Observer*)

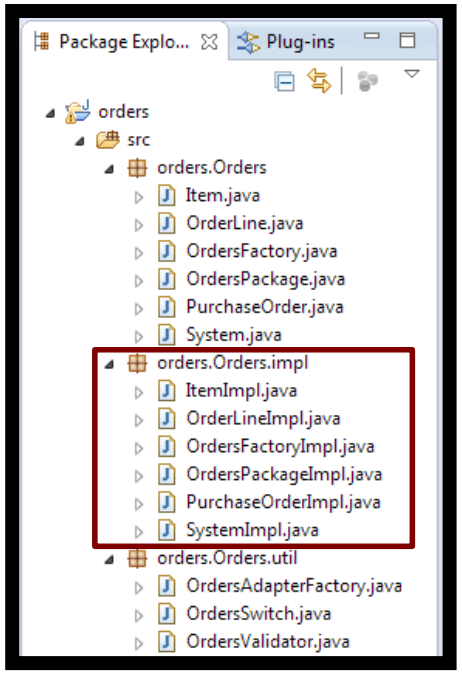


Clases Java generadas

Clases de implementación de las interfaces

Cómo usar las clases generadas:

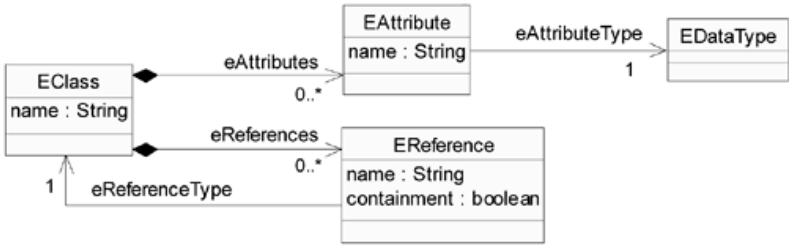
```
System mediamarkt = OrdersFactory.eINSTANCE.createSystem();

Item cable = OrdersFactory.eINSTANCE.createItem();
cable.setName ("Cable VGA");
cable.setPrice(BigDecimal.valueOf(15));
cable.setStock(BigInteger.valueOf(250));

mediamarkt.getStock().add(cable);
```



Podemos añadir métodos a las clases generadas.

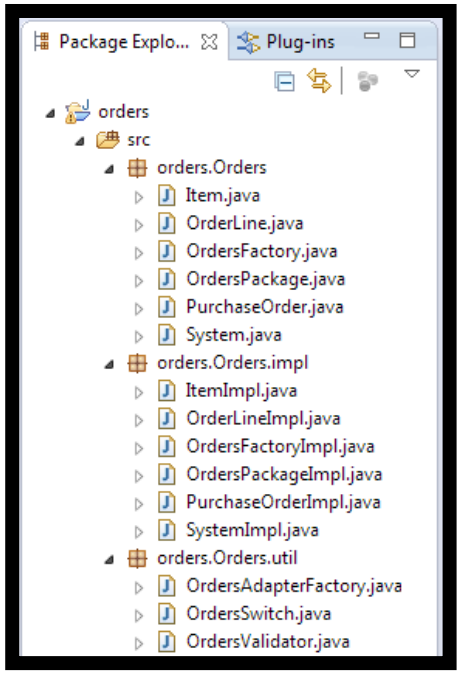
Los métodos generados tienen la anotación `@generated`. Si se modifican a mano, quitar la anotación para evitar que se sobreescriban al regenerar el código.

Clases Java generadas
EMF dinámico



Es posible acceder a la definición de los objetos reflexivamente en tiempo de ejecución:

```
Item cable = OrdersFactory.eINSTANCE.createItem();

EClass clase = cable.eClass();
List<EAttribute> atributos = clase.getEAllAttributes();
for (EAttribute atr : atributos) {
  System.out.println( atr.getName() );
  System.out.println( atr.getLowerBound() );
  System.out.println( atr.getUpperBound() );
  System.out.println( atr.getEType().getName() );
}
```



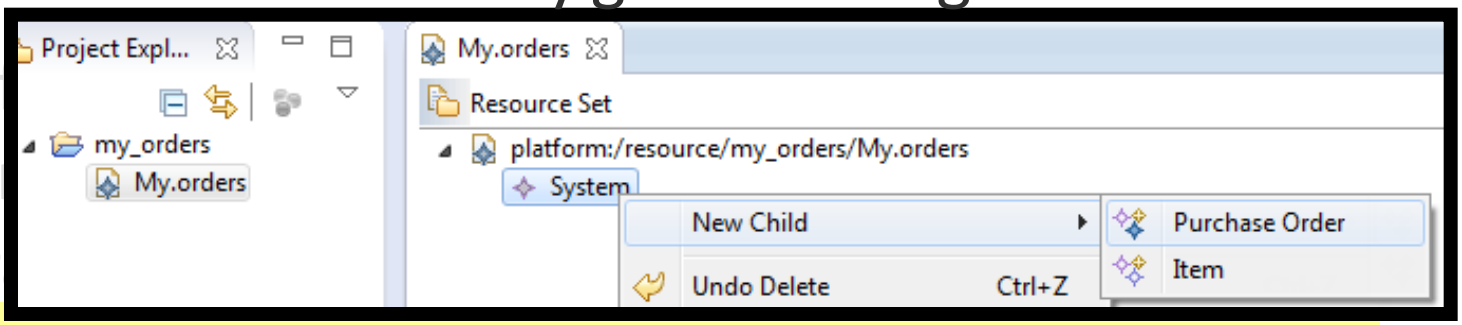
También se pueden crear meta-modelos ecore dinámicamente, e instanciarlos:

https://www.ibm.com/developerworks/library/os-eclipse-dynamicemf/

Eclipse Modeling Framework

Cómo definir un meta-modelo y generar código

1. Crear
2. Crear
3. Gener



4. **Generar editor de modelos en forma de árbol**
   - Abrir modelo generador, botón derecho en raíz
     Generate Edit Code, y a continuación, *Generate Editor Code*
   - Lanzar segunda instancia de Eclipse
     Run / Run As / Eclipse Application
     (despliega el editor, que es un plugin, en la nueva instancia de Eclipse)
   - Crear proyecto vacío en la nueva instancia de Eclipse
   - Crear modelo en el proyecto
     File / New / Other / Example EMF Model Creation Wizards / new language
     (deberás seleccionar la clase raíz del modelo, `System` en este ejemplo)

Bibliografía

- EMF: Eclipse Modeling Framework. Dave Steinberg, Frank Budinski, Marcelo Paternostro & Ed Merks. Addison-Wesley Professional, 2nd Edition (2008)

- Model-Driven Software Engineering in Practice (Synthesis Lectures on Software Engineering). Marco Brambilla, Jordi Cabot & Manuel Wimmer. Morgan & Claypool Publishers, 1st Edition (2012)