

**MODEL-DRIVEN
SOFTWARE DEVELOPMENT**

FOR
DUMMIES®

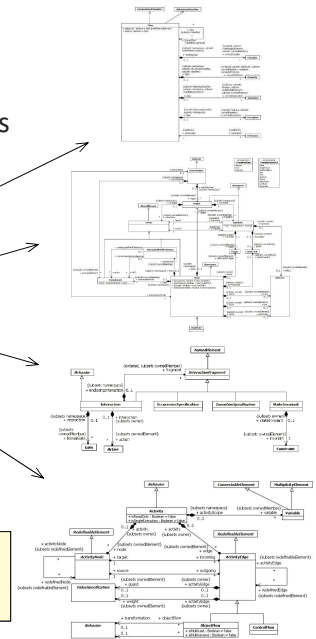
SLAY THE
COMPLEXITY DRAGON
WITH ABSTRACTION
AND AUTOMATION

*A Reference
for the
Rest of Us!*

FREE e-Chapter at www.it-ebooks.info

1

EMF



```

graph BT
    MM[Meta model MM]
    M[Model M]
    M -- "instance of" --> MM
    MM -- "conforms to" --> MM
  
```

- 3

4

Eclipse Modeling Framework

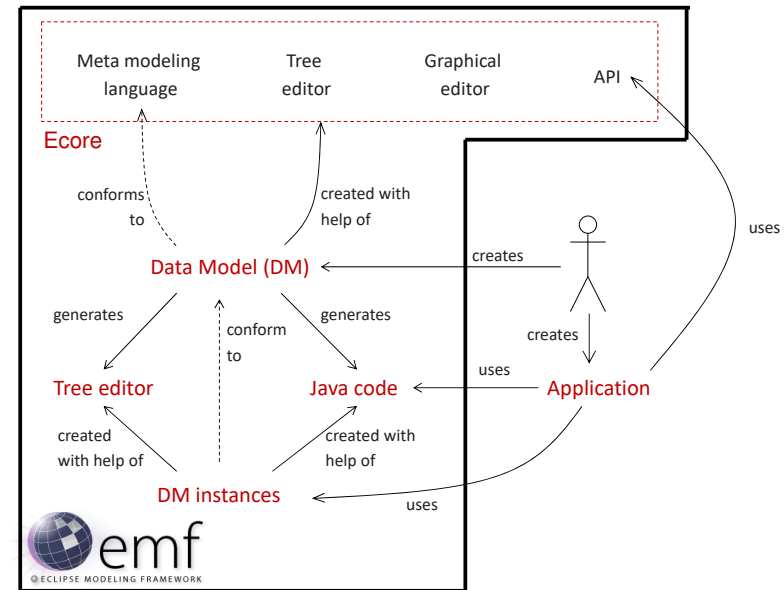


- Eclipse-based open-source framework supporting the development of applications requiring a structured **data model (DM)**
- Consists of
 - Ecore**
 - Meta modelling language for the description of DMs with support for
 - change notification (via the Observer design pattern),
 - persistence (via XML serialization), and
 - generic object manipulation (via reflective API)
 - EMF.Codegen**
 - Generates code implementing the DM and supporting the development of tools creating and manipulating data model instances:
 - Model**: Java interfaces and implementation of DM (using Factory design pattern)
 - Editor**: plugin for tree-based editor of DM instances
 - Edit**: implementation classes adapting DM classes for editing and display (via Adapter Factory design pattern)
 - Large user community**: <http://www.eclipse.org/emf>

CISC836, Fall 2021

EMF

5



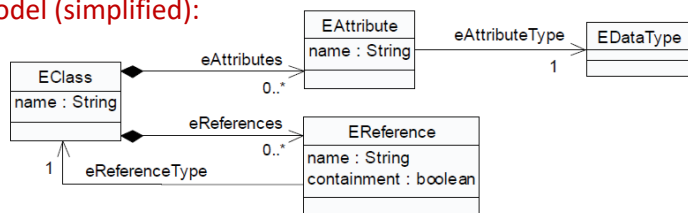
CISC836, Fall 2021

EMF

6

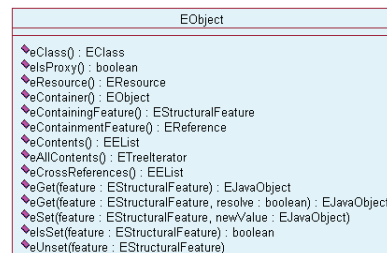
Ecore: Metamodeling Language

- Metamodel (simplified):**



- EObject**

- Base of every Ecore class and every generated class
- Provides support for **notification** and **persistence**



[Summary of package org.eclipse.emf.ecore]

[Steinberg, Budinsky, Paternostro, Merks. EMF: Eclipse Modeling Framework (2nd Ed.). 2008]

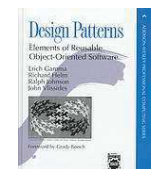
CISC836, Fall 2021

EMF

7

EMF: Supporting Technology

- Eclipse**
 - Editors are Eclipse plugins
- Sirius**
 - Framework for automatic generation of graphical editors
- JUnit**
 - Supported by generated test code
- Design patterns**
 - Observer
 - Factory
 - Adapter Factory



CISC836, Fall 2021

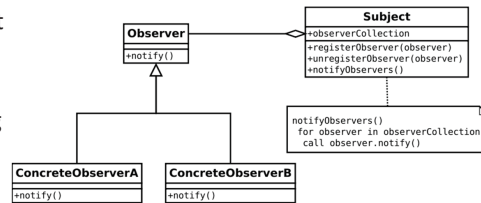
EMF

8

Observer Design Pattern

Object, called **subject**, maintains list of its dependents, called **observers**, and **notifies them automatically of any state changes**, usually by calling one of their methods.

Used for **MVC pattern**.



```

import java.util.Observable;
import static java.lang.System.out;

class MyApp {
    public static void main(String[] args) {
        out.println("Enter Text >");
        EventSource eventSource = new EventSource();
        eventSource.addObserver((Observable obj, Object arg) -> {
            out.println("\nReceived response: " + arg);
        });
        new Thread(eventSource).start();
    }
}

import java.util.Observable;
import java.util.Scanner;

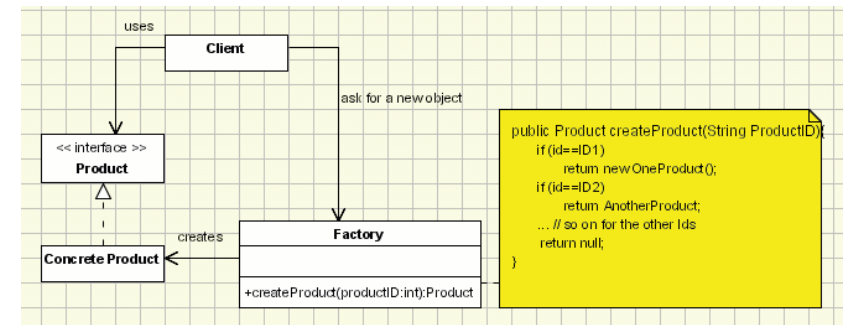
class EventSource extends Observable implements Runnable {
    public void run() {
        while (true) {
            String response = new Scanner(System.in).next();
            setChanged();
            notifyObservers(response);
        }
    }
}

```

[Wikipedia. Observer pattern. https://en.wikipedia.org/wiki/Observer_pattern. 2016]

Factory Design Pattern

Object, called the **Factory**, encapsulates details of creation of different **Products**



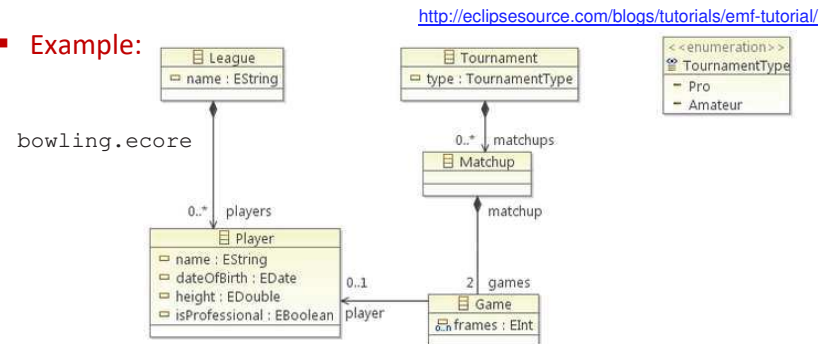
[OODesign.com. Factory pattern. <http://www.oodesign.com/factory-pattern.html>. 2016]

Using EMF: Getting Started

- Download and installation
 - <http://www.eclipse.org/modeling/emf>
- Documentation
 - <http://www.eclipse.org/modeling/emf/docs/>
- Tutorials
 - <http://www.vogella.com/tutorials/EclipseEMF/article.html>
 - <http://eclipsesource.com/blogs/tutorials/emf-tutorial/>
 - <http://www.philmann-dark.de/EMFDocs/tutorial.html>

Using EMF: Create Domain Model (DM)

- Example:



- Root classes: League, Tournament
- Containment relationship b/w Matchup and Game is **bi-directional**
 - "Containment = true" for reference games
 - matchup is opposite of games:
 - for all **m:Matchup**, if **g:Game** in **m.games**, then **g.matchup == m**

EMF Sample Models



Beyond Code: An Introduction to Model-Driven Software Development (CISC 836, Winter 2021)

Sample Eclipse Modeling Framework (EMF) Projects

The sample projects below assume the Eclipse IDE for Java and DSL Developers (version 2020-12 R; it will contain EMF version 2.24). See the beginning of [Assignment 4](#) for information on how to obtain and install it. The origin of these projects is as follows:

- **Extended library, Java:** Part of the EMF distribution
- **Bowling:** Running example of the EclipseSource EMF tutorial [here](#)
- **Linked lists:** C++

All artifacts are made available under the terms of the [Eclipse Public License v1.0](#).

1. Extended library:

- class (data) model for a library
- illustrates understanding with ECore (concepts: classes (concrete, abstract, subclasses), associations (multiplicities, inverses), attributes (types, defaults, multiplicities, uniqueness, unchangeable), enumerations), editor plugin generation
- ECore metamodel:



EMF Sample Models

CISC836, Fall 2021

EMF

13

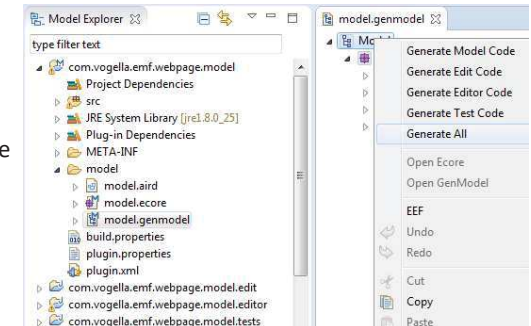
Using EMF: Generate Code from DM

■ Genmodel

- Generated from .ecore model
- Contains additional info necessary for code generation (package names, source paths, project names, generator settings)
- Automatically synchronized with .ecore model when .ecore model is saved

■ To generate:

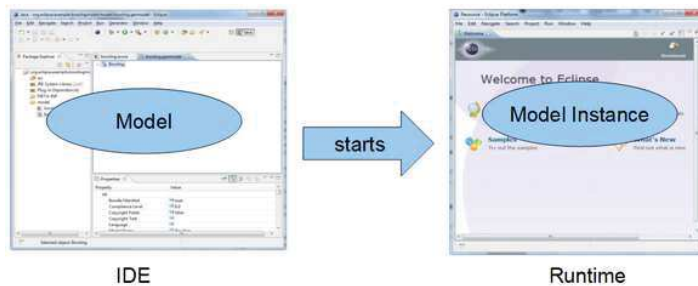
- Open .genmodel
- right-click root node
- select plugin to generate



CISC836, Fall 2021

Using EMF: Use Generated DM Editor

- Code for DM editor generated as Eclipse plugin
- Invocation of the editor will create another instance of Eclipse in which the editor will execute

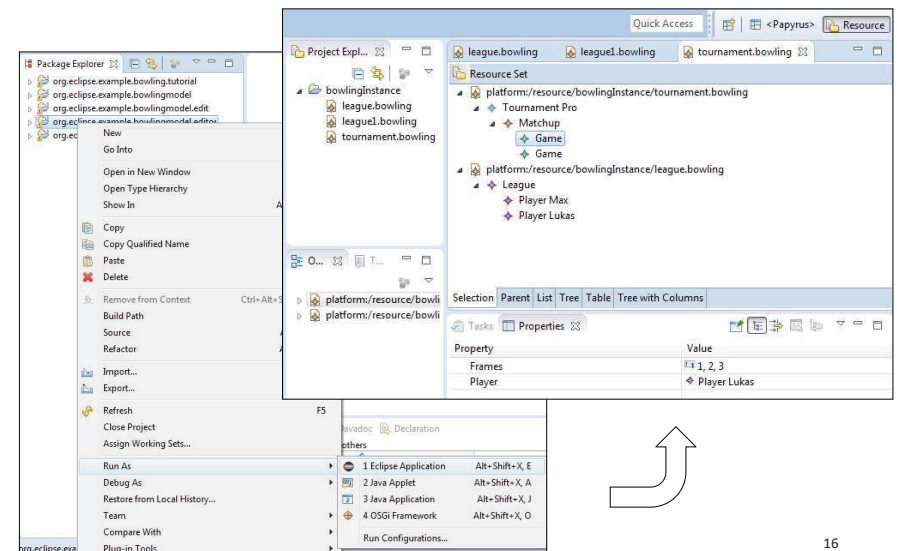


CISC836, Fall 2021

EMF

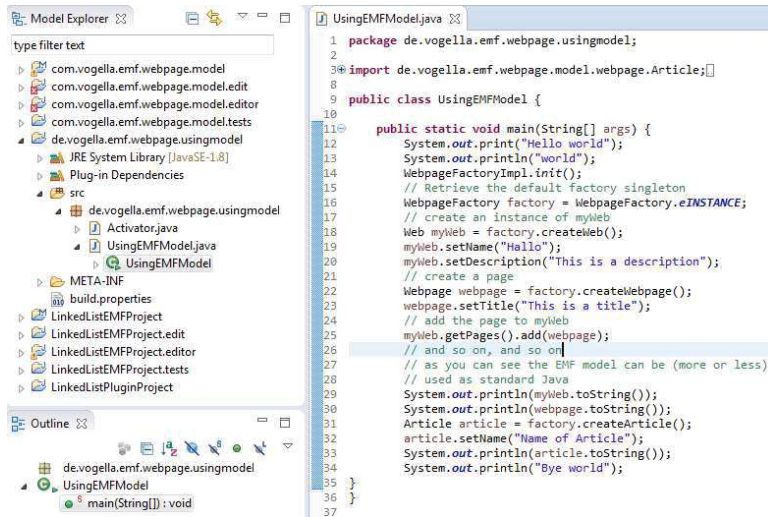
15

Using EMF: Use Generated DM Editor (Cont'd)



16

Using EMF: Use Generated DM Java Code



CISC836, Fall 2021

EMF

17

Using EMF: Demo

- Using generated test code
- Generating JavaDoc
- Generating method bodies



CISC836, Fall 2021

EMF

18

EMF: Pros and Cons

- Pros
 - Quite powerful, stable, adequate documentation
 - Integral part of Eclipse Modeling ecosystem:
 - EMF Forms: for generation of form-based UI
 - Xtext: for DSL implementation
 - ATL: for model-to-model transformation
- Cons
 - Inherit Eclipse “baggage”

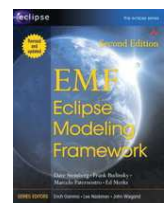
CISC836, Fall 2021

EMF

19

EMF: More Info

- Installation
 - <http://www.eclipse.org/modeling/emf>
- Documentation
 - <http://www.eclipse.org/modeling/emf/docs/>
- Tutorials
 - <http://eclipsesource.com/blogs/tutorials/emf-tutorial/>
 - <http://www.vogella.com/tutorials/EclipseEMF/article.html>
 - <http://www.philmann-dark.de/EMFDocs/tutorial.html>
- Book
 - Steinberg, Budinsky, Paternostro, Merks. EMF: Eclipse Modeling Framework (2nd Ed.). Addison-Wesley Professional. 2008.



CISC836, Fall 2021

EMF