What the heck is **Eclipse Modeling** and why should **you** care !

Eclipse Day Paris

Paris, 8 Nov. 2011

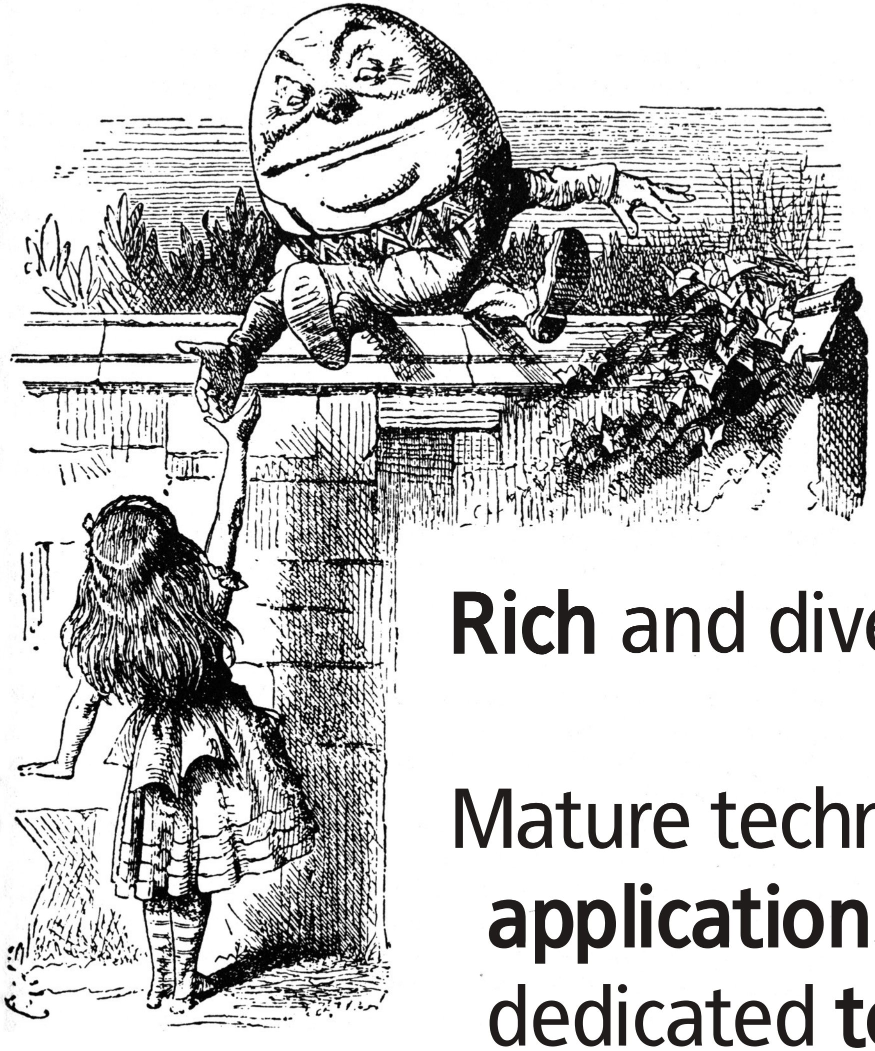
me

CTO at Obeo – cedric.brun@obeo.fr
Strategic Member of the Eclipse Foundation

Projet lead of:
- EMF Compare
- Modeling Amalgamation

Commiter:
- Acceleo
- Mylyn Intent
- Eclipse Packaging Project

Eclipse Architecture and Planning Council



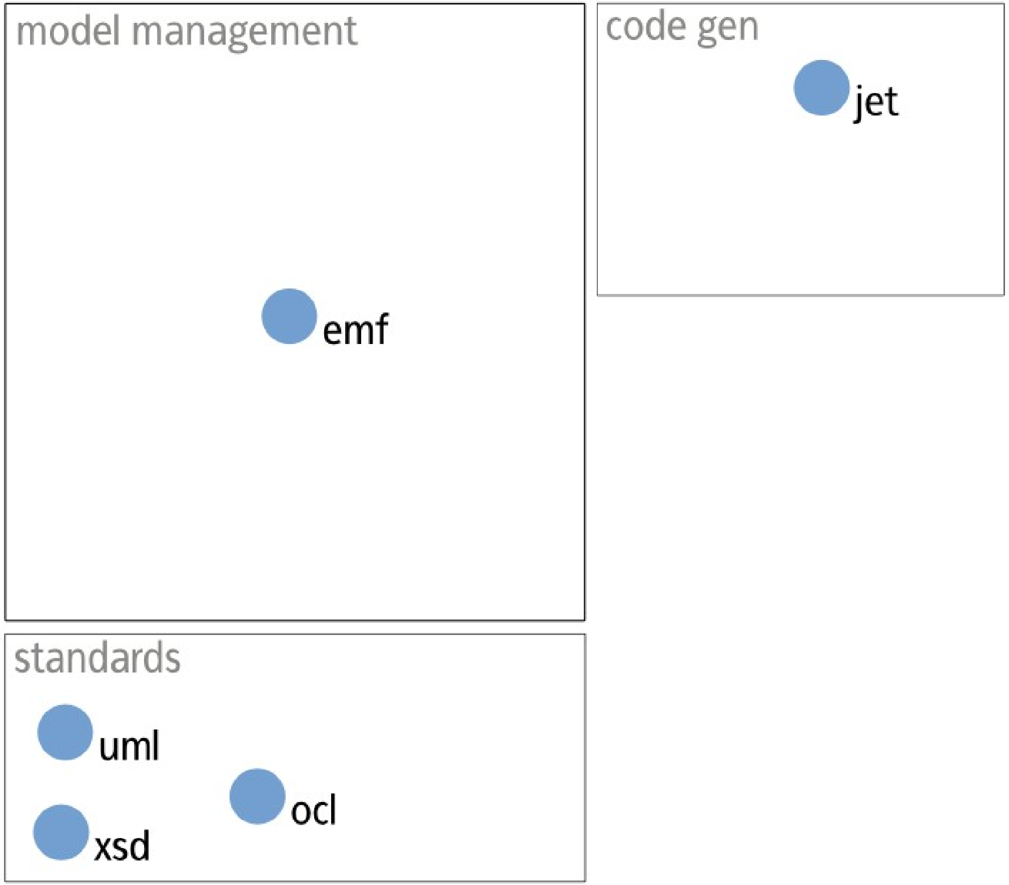
Rich and diverse Ecosystem

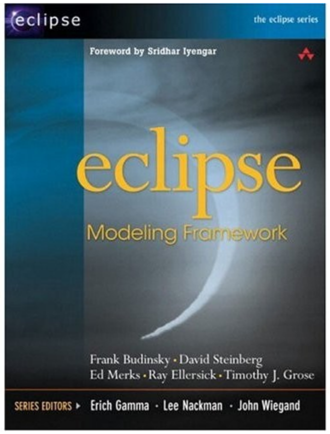
Mature technologies and frameworks to build
- **applications**
- dedicated **tools**

You can try at a very **low cost**

Happy **birthday**

The origins




eclipse

the eclipse series

Foreword by Sridhar Iyengar

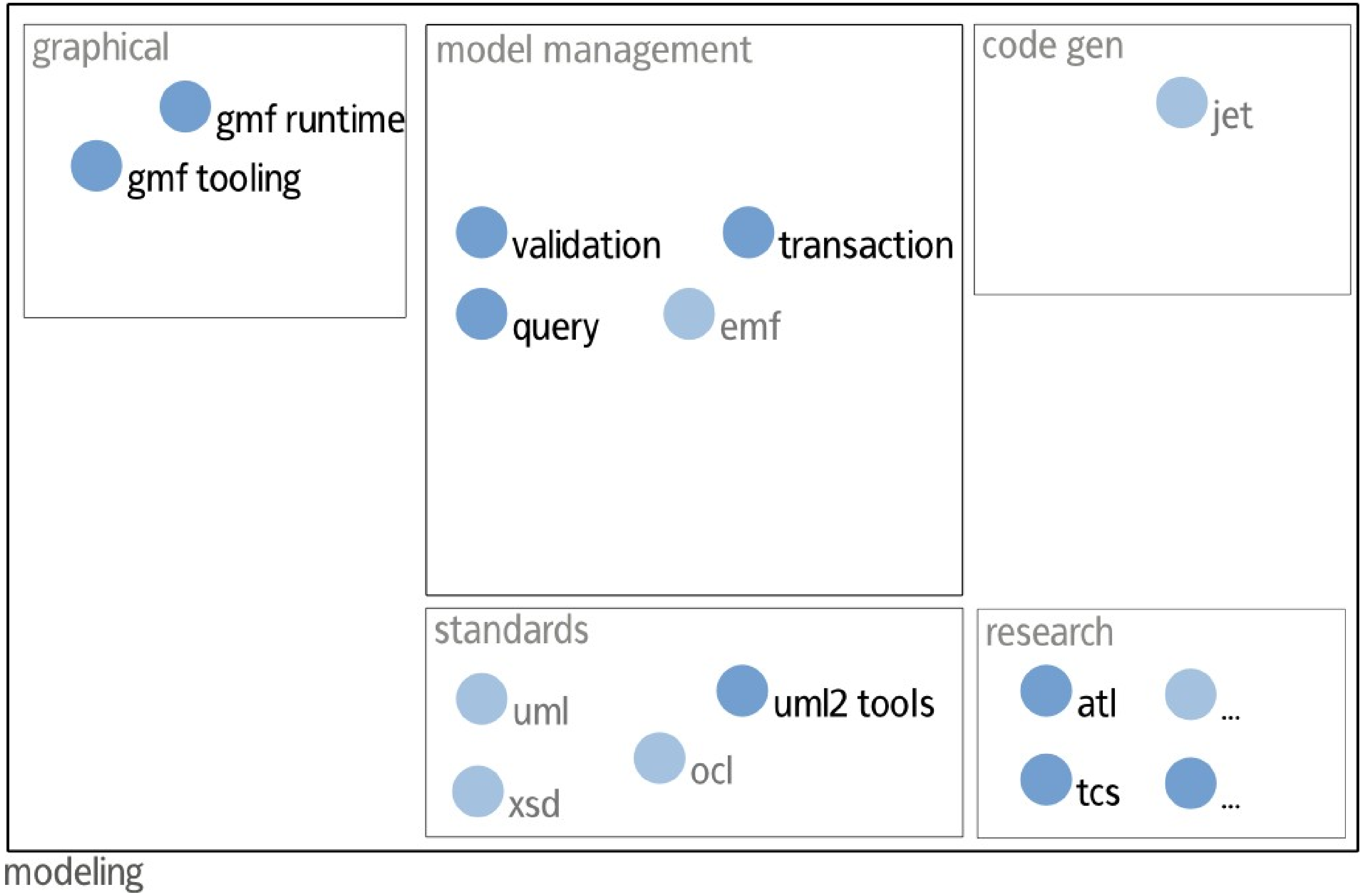
eclipse

Modeling Framework

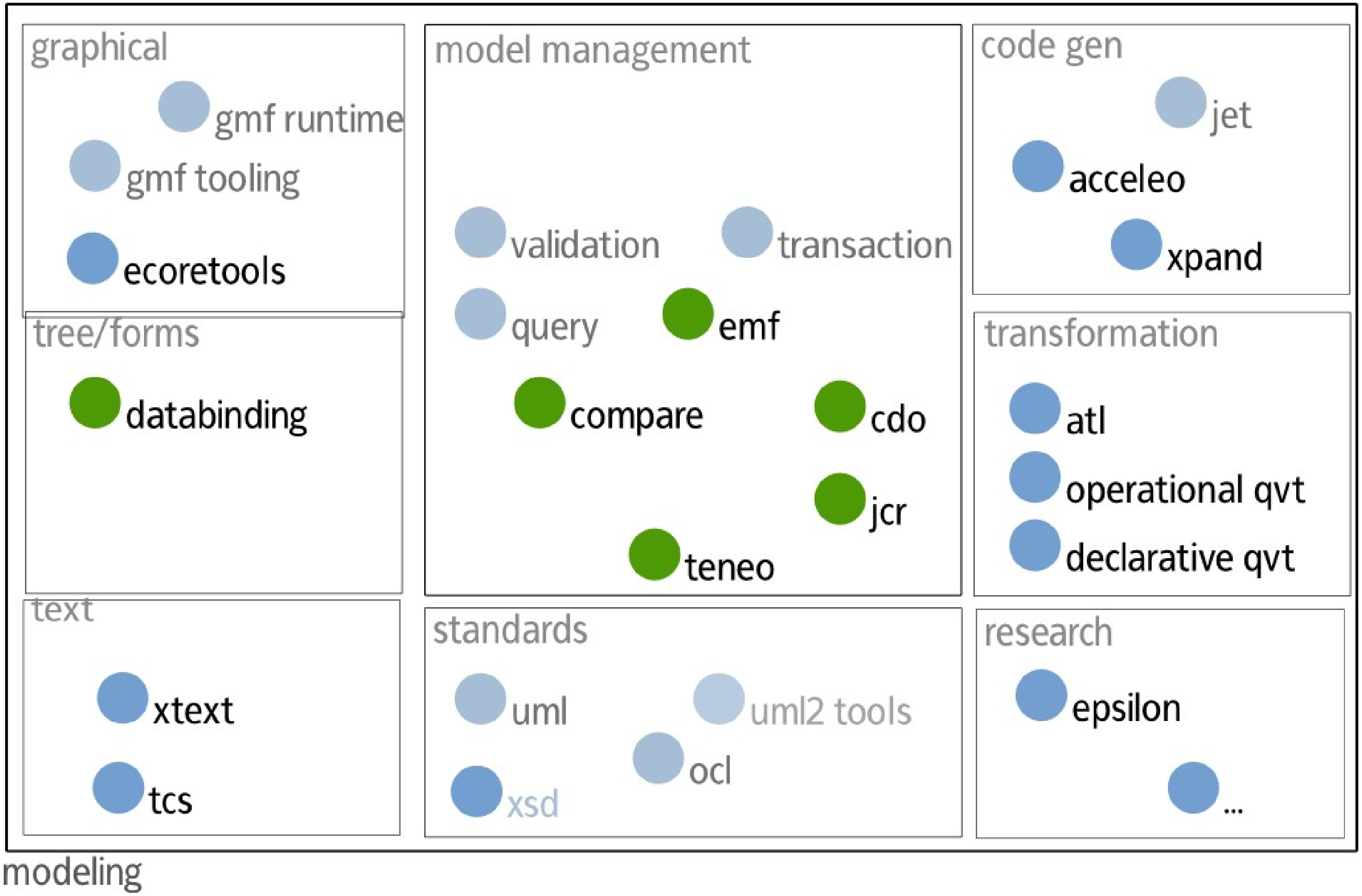
Frank Budinsky · David Steinberg
Ed Merks · Ray Ellersick · Timothy J. Grose

SERIES EDITORS ▸ Erich Gamma · Lee Nackman · John Wiegand

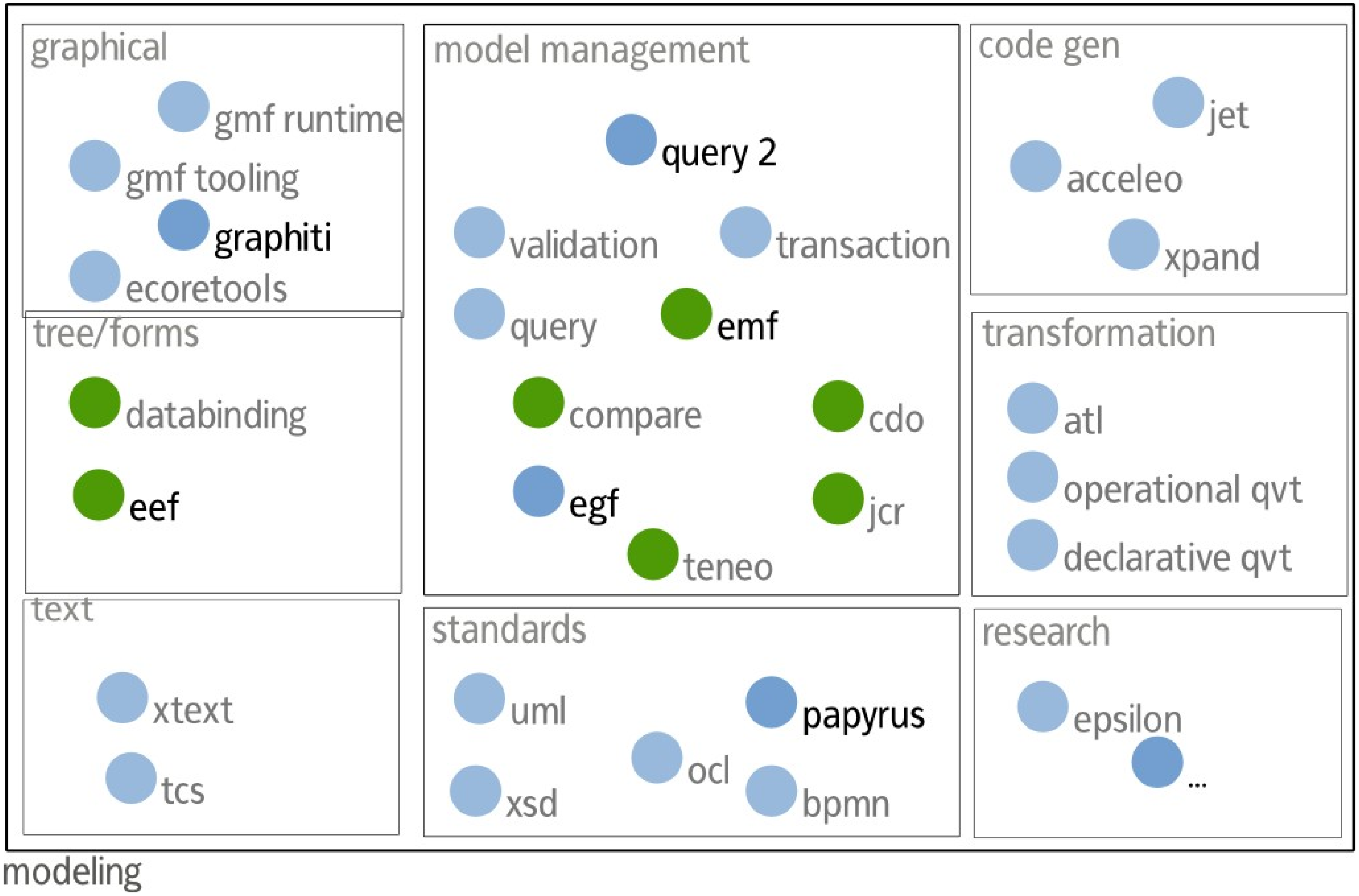
Tool vendors era



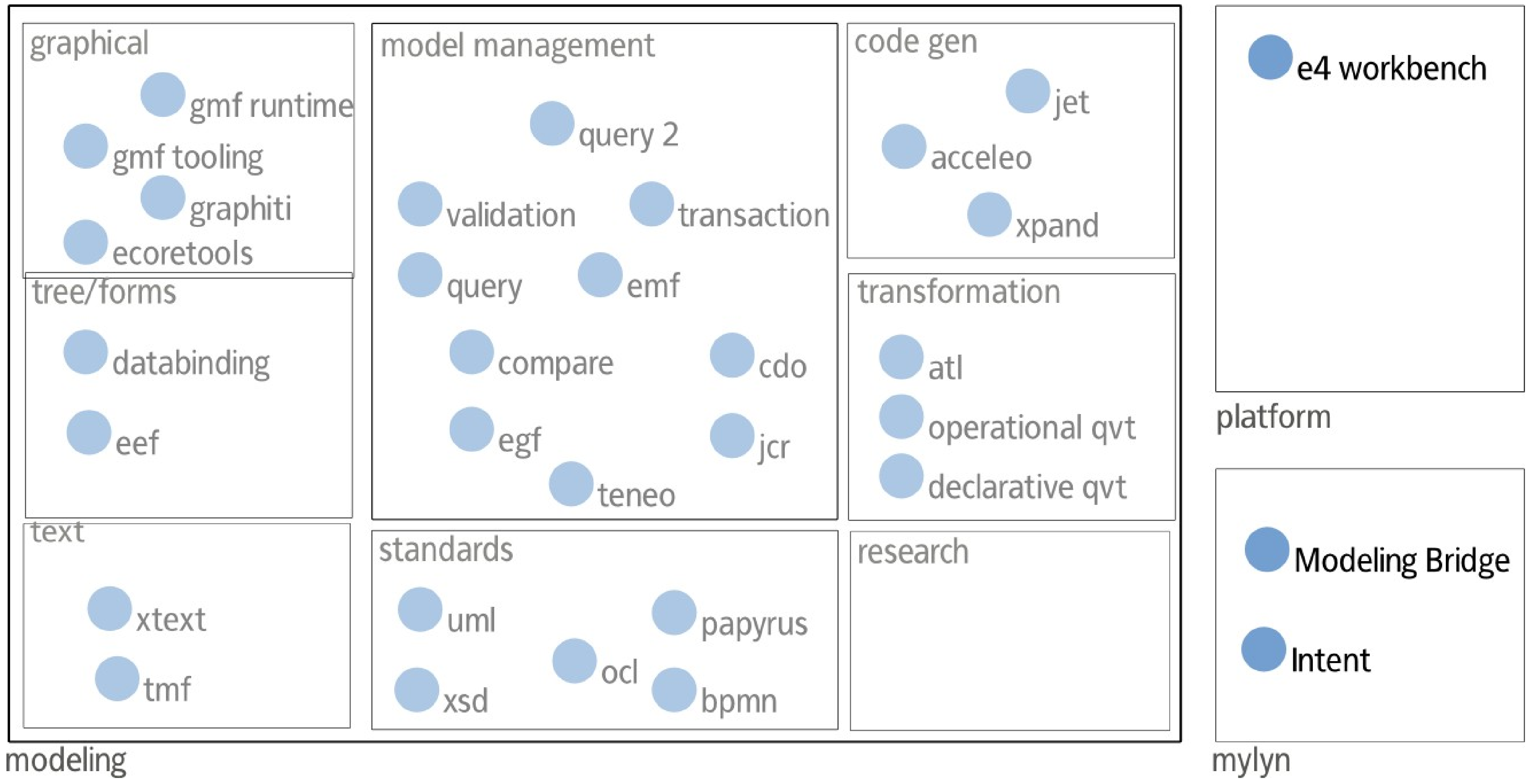
Modeling goes enterprise !



Industrial adopters



Resistance is futile



graphical

gmf runtime

gmf tooling

graphiti

ecoretools

tree/forms

databinding

eef

text

xtext

tmf

model management

query 2

validation

transaction

query

emf

compare

cdo

egf

jcr

teneo

standards

uml

papyrus

ocl

xsd

bpmn

code gen

jet

acceleo

xpand

transformation

atl

operational qvt

declarative qvt

research

modeling

e4 workbench

platform

Modeling Bridge

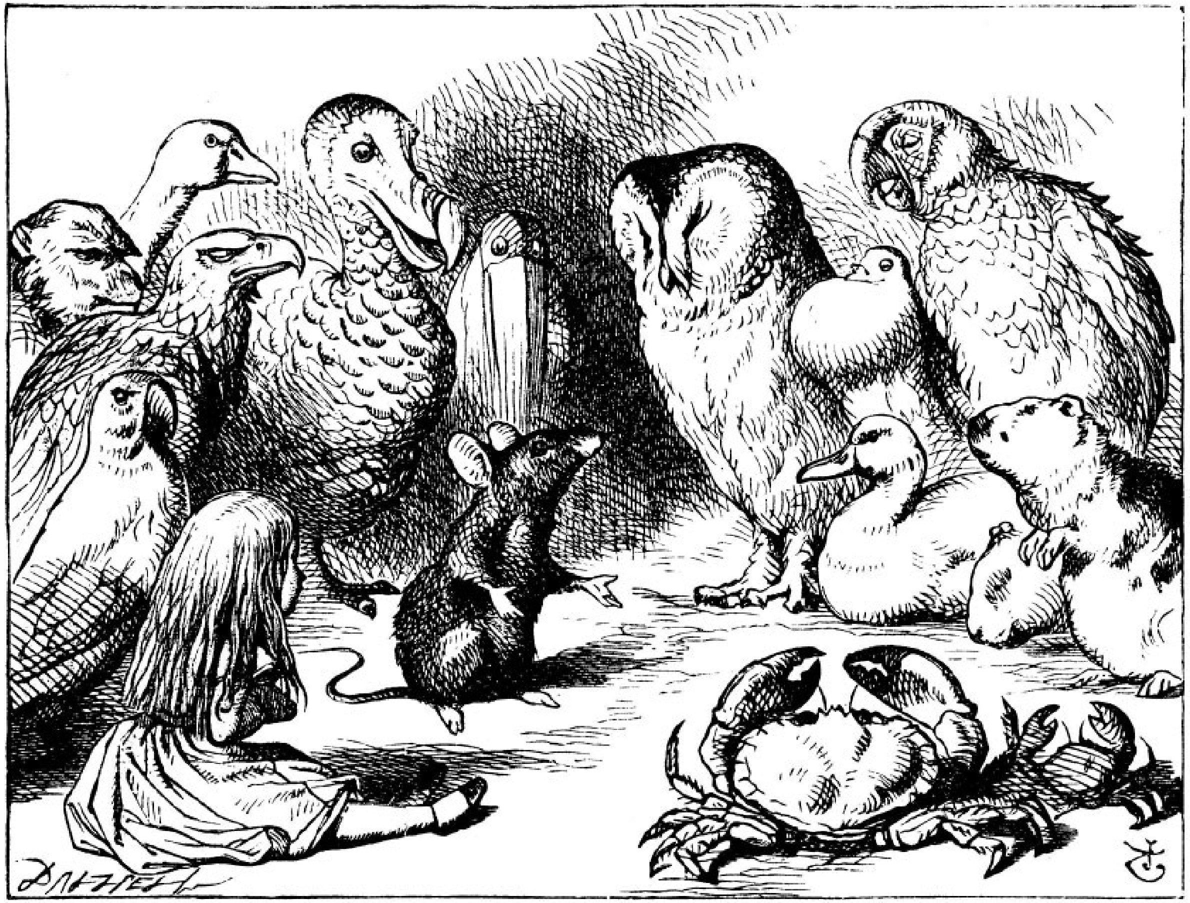
Intent

mylyn

The Eclipse Modeling Project

One of the most diverse **top-level** project in Eclipse

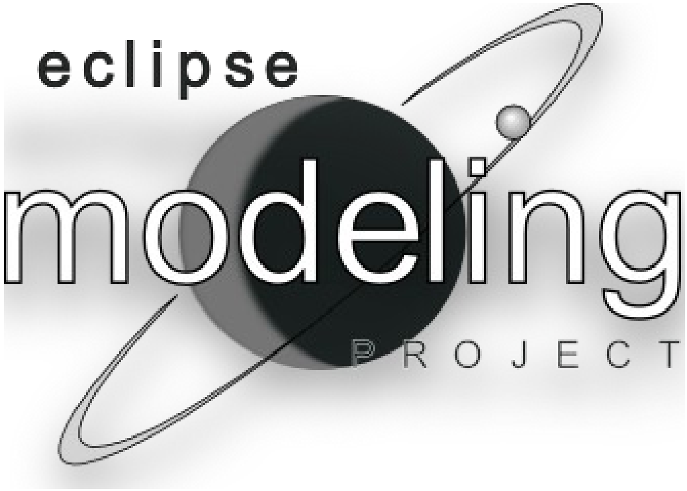
- Tool vendors
- Service providers
- Individuals
- Academics



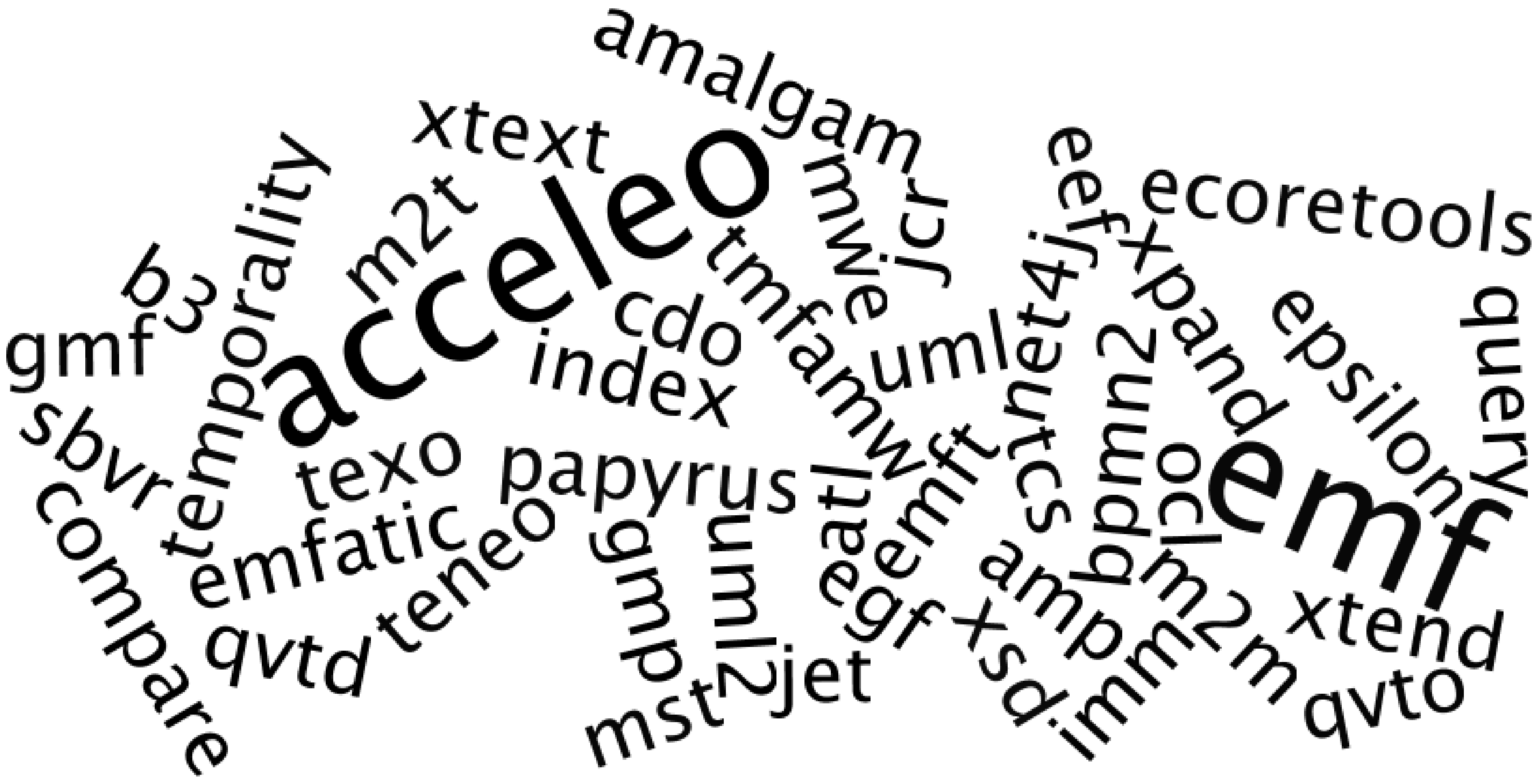
The Eclipse Modeling Project

- Wide range of public

- Wide range of maturity level
    - **Ask for help on the forum**
    - Check for activity
    - Mind the Incubation status



The most amazing **Acronym** collection



The Eclipse Modeling Project
Structure

A «few» sub projects:

- Core technologies in **EMF**, Incubating in **EMFT**
- Standards and related tools in **MDT**
- Graphical Modeling in **GMP**
- Textual Modeling in **TMF**
- Model Transformation in **M2M**
- Code Generation in **M2T**
- Agent Based Modeling and Simulation in **AMP**

And

- Integration, Packaging & Examples in **Amalgamation**

we **Deliver**

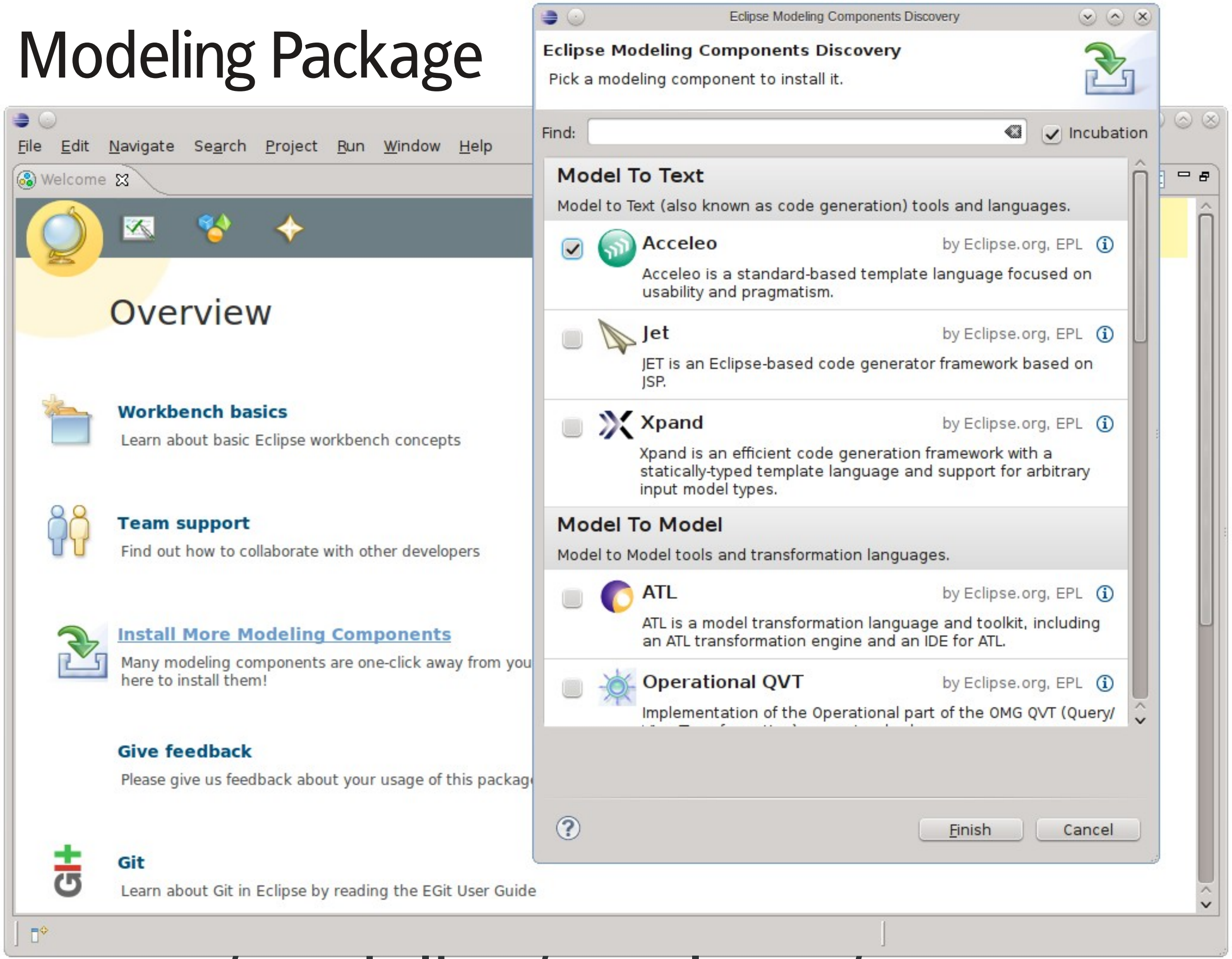
Most projects are part of the release trains

Galileo, Helios, Indigo and soon Juno

A single stop: the Eclipse Modeling Package

Core platform

Discovery



Workbench basics
Learn about basic Eclipse workbench concepts

Team support
Find out how to collaborate with other developers

Install More Modeling Components
Many modeling components are one-click away from you
here to install them!

Give feedback
Please give us feedback about your usage of this packag

Git
Learn about Git in Eclipse by reading the EGit User Guide

Eclipse Modeling Components Discovery
Pick a modeling component to install it.

Model To Text
Model to Text (also known as code generation) tools and languages.

☑ **Acceleo** by Eclipse.org, EPL
Acceleo is a standard-based template language focused on usability and pragmatism.

□ **Jet** by Eclipse.org, EPL
JET is an Eclipse-based code generator framework based on JSP.

□ **Xpand** by Eclipse.org, EPL
Xpand is an efficient code generation framework with a statically-typed template language and support for arbitrary input model types.

Model To Model
Model to Model tools and transformation languages.

□ **ATL** by Eclipse.org, EPL
ATL is a model transformation language and toolkit, including an ATL transformation engine and an IDE for ATL.

□ **Operational QVT** by Eclipse.org, EPL
Implementation of the Operational part of the OMG QVT (Query/

http://www.eclipse.org/modeling/amalgam/

What
is Modeling ?

Capturing information to represent a system enabling:

Validation
Prototyping
Automation
Documentation
...

What makes a **good** model ?

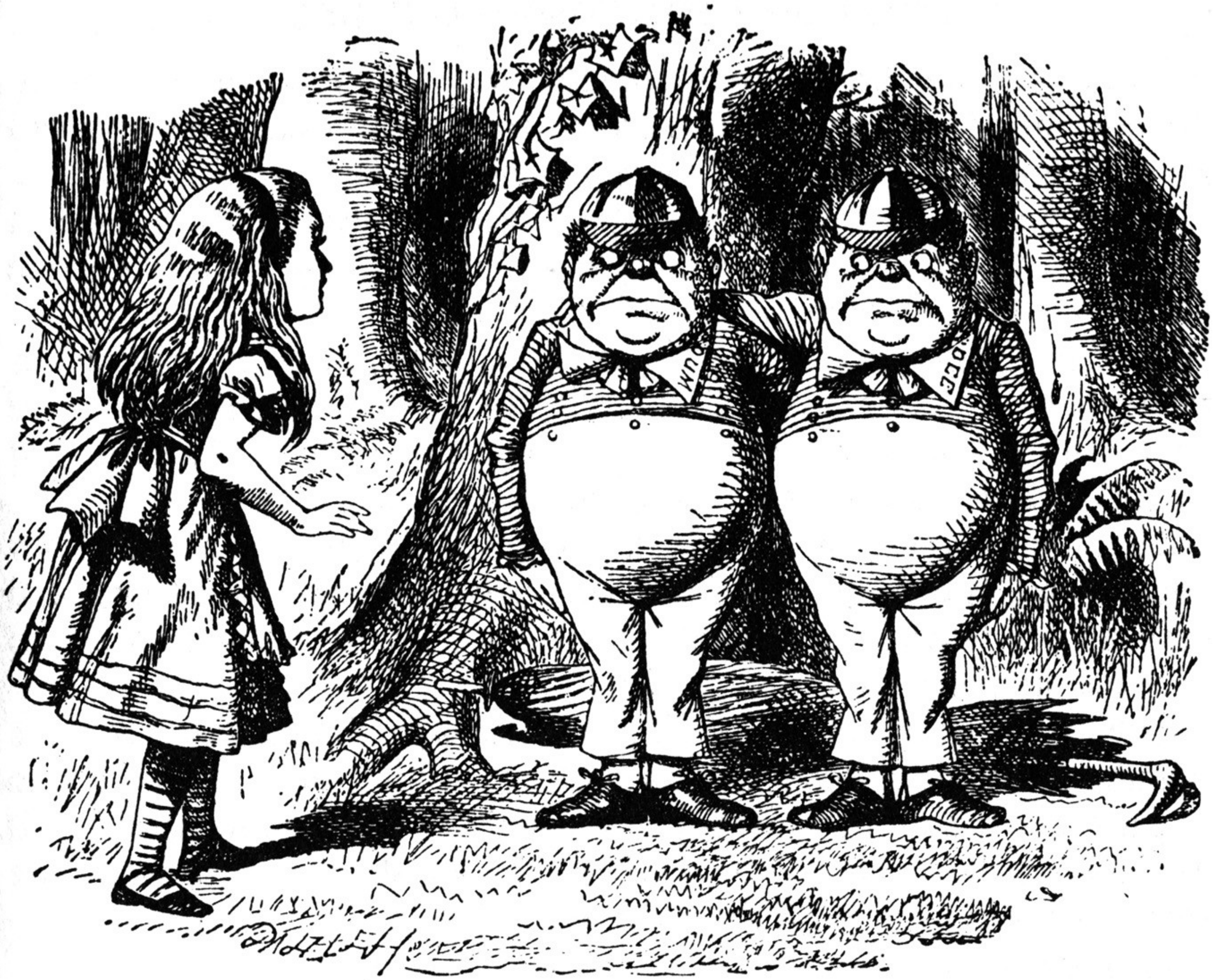
1. It is answering a question

2. It is used

Two Faces

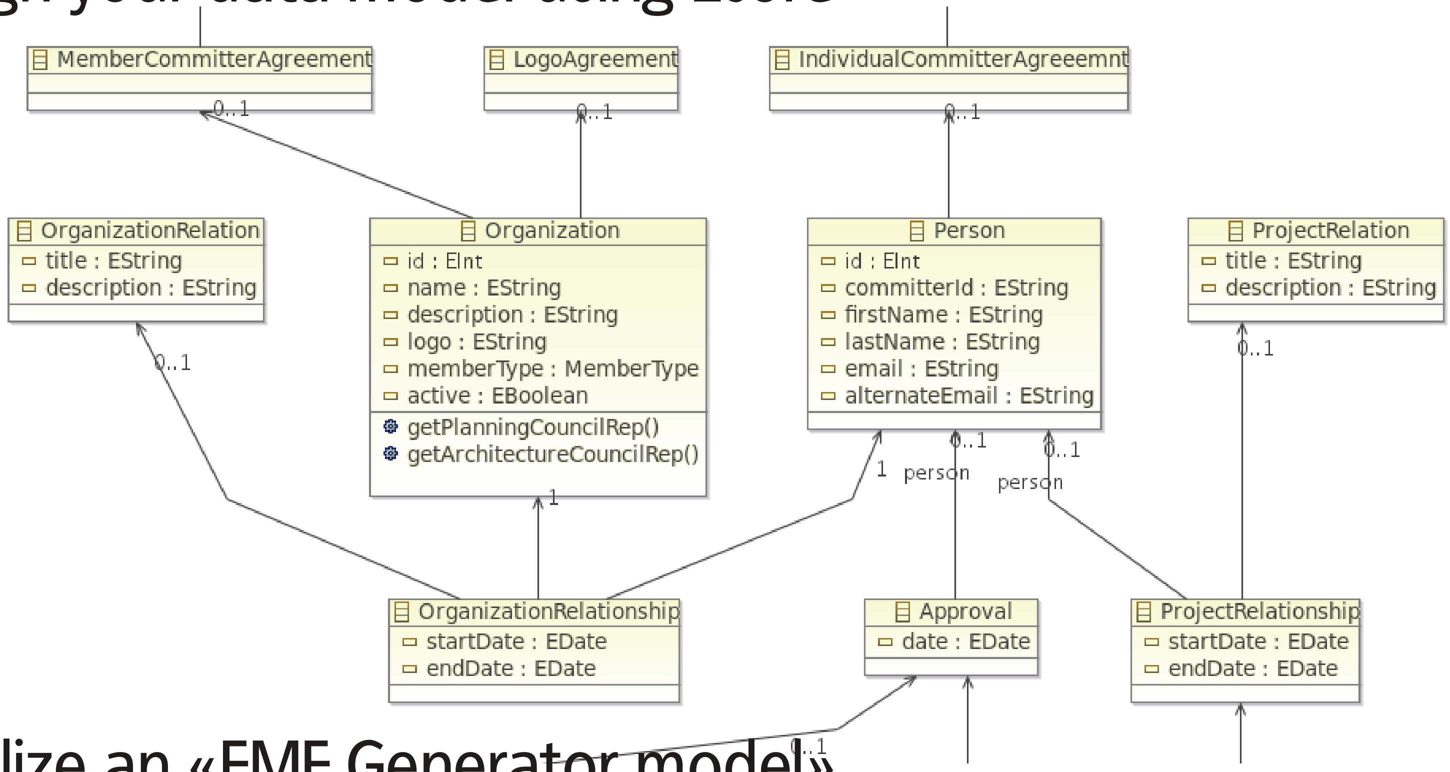
➡ Technologies to build **applications**

Technologies to build **dedicated tools**



Using EMF **«as is»**

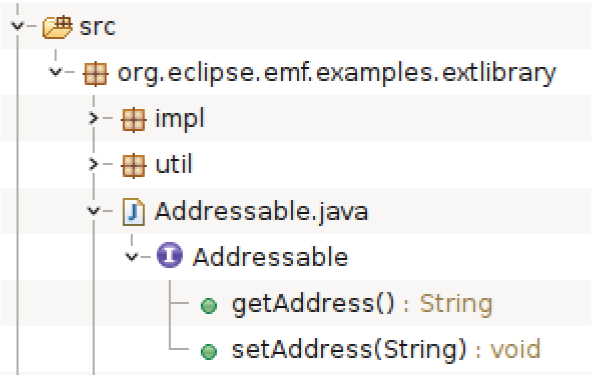
1° Design your data model using Ecore



2° Initialize an «EMF Generator model»

extlibrary.genmodel

3° Generate Java code



- src
  - org.eclipse.emf.examples.extlibrary
    - impl
    - util
    - Addressable.java
      - Addressable
        - getAddress() : String
        - setAddress(String) : void

Using EMF **«as is»**

You are getting:

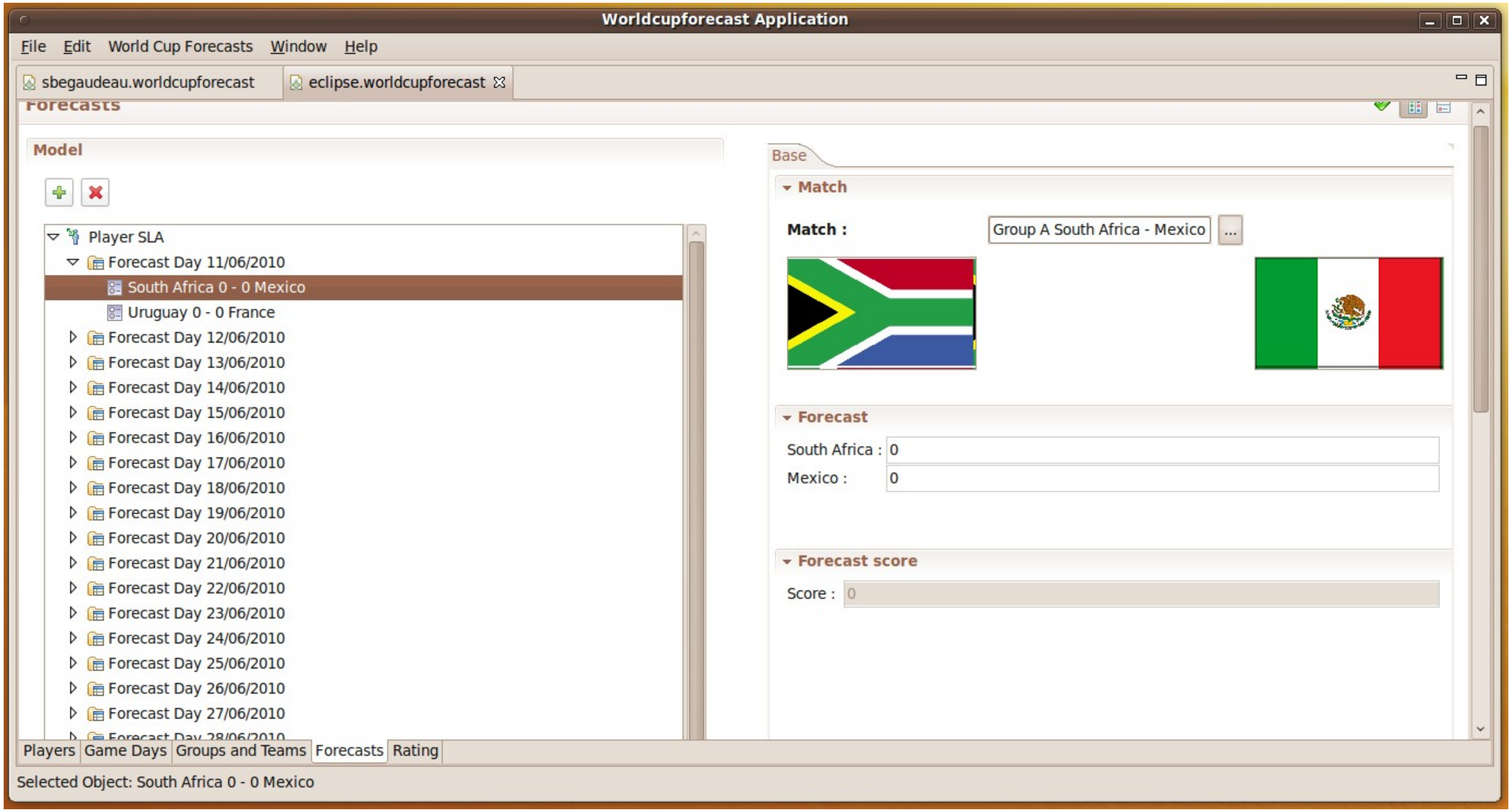
- ✓ **Domain model** independent of serialization (XMI, XML, Binary ...)
- ✓ **Efficient Java** API
- ✓ Referenceable model elements through **URI + fragments**
- ✓ Common API to **listen** and react to changes.
- ✓ Resources and **REST** api for documents
- ✓ Rich and efficient reflective API.
- ✓ Commands and free **undo/redo**
- ✓ ...

At a cost:

- ✓You'll **depend on EMF** (which is standalone, but still...)

You need **UI**, right ?

EMF and EEF are generating rich applications with forms



- Player SLA
  - Forecast Day 11/06/2010
    - South Africa 0 - 0 Mexico
    - Uruguay 0 - 0 France
  - Forecast Day 12/06/2010
  - Forecast Day 13/06/2010
  - Forecast Day 14/06/2010
  - Forecast Day 15/06/2010
  - Forecast Day 16/06/2010
  - Forecast Day 17/06/2010
  - Forecast Day 18/06/2010
  - Forecast Day 19/06/2010
  - Forecast Day 20/06/2010
  - Forecast Day 21/06/2010
  - Forecast Day 22/06/2010
  - Forecast Day 23/06/2010
  - Forecast Day 24/06/2010
  - Forecast Day 25/06/2010
  - Forecast Day 26/06/2010
  - Forecast Day 27/06/2010
  - Forecast Day 28/06/2010

Match

Match : Group A South Africa - Mexico

Forecast

South Africa : 0

Mexico : 0

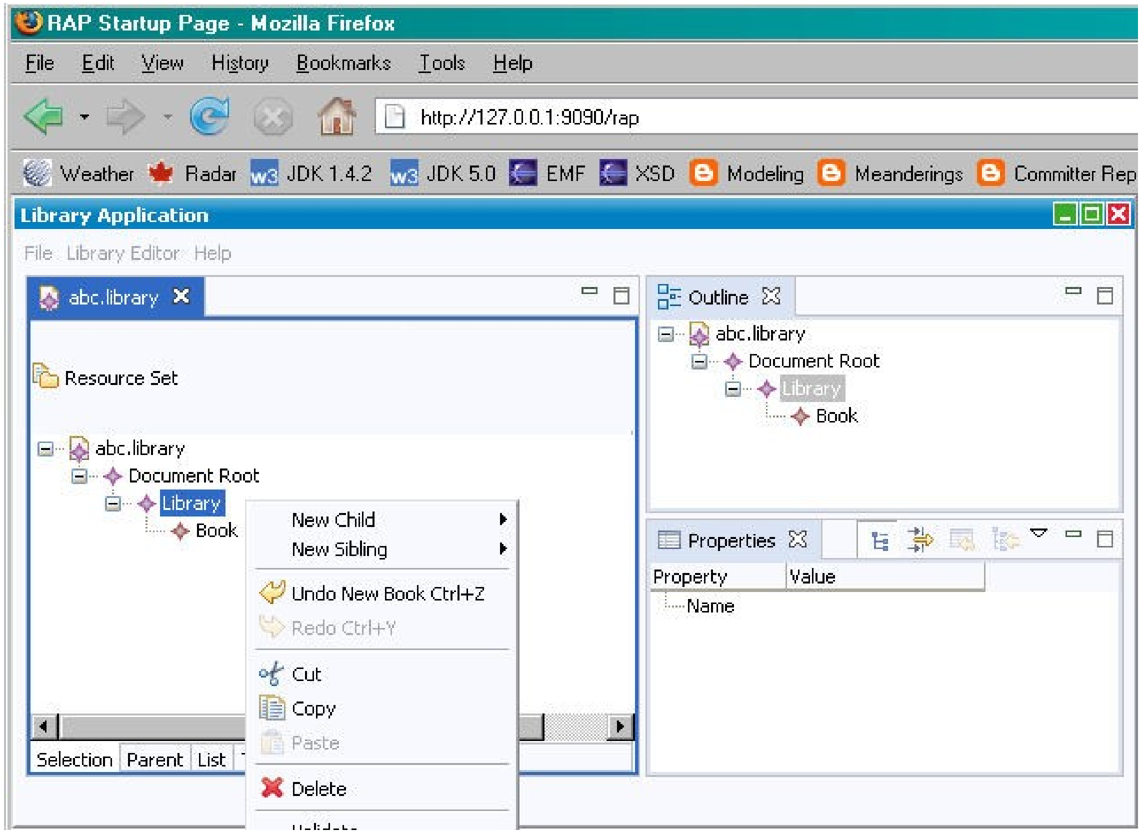
Forecast score

Score : 0

Selected Object: South Africa 0 - 0 Mexico

Generated web apps

EMF generates RAP application (RCP in a browser)



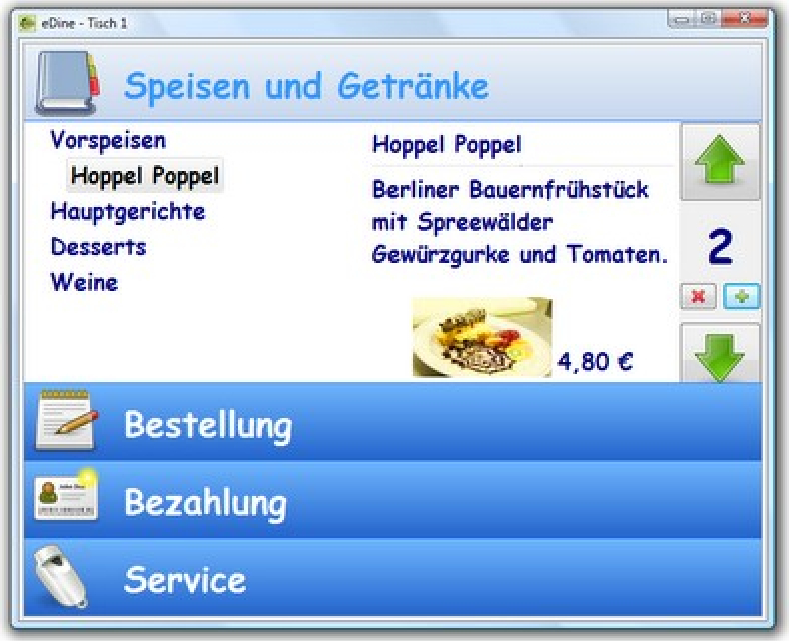
Or GWT !

+ Model Repository

With CDO :

- ✓ Distributed and connected application
- ✓ Optimistic and pessimistic concurrent strategies
- ✓ Offline support
- ✓ Lazy loading, and **unloading** !
- ✓ Database backend
- ✓ Fail-over
- ✓ ...

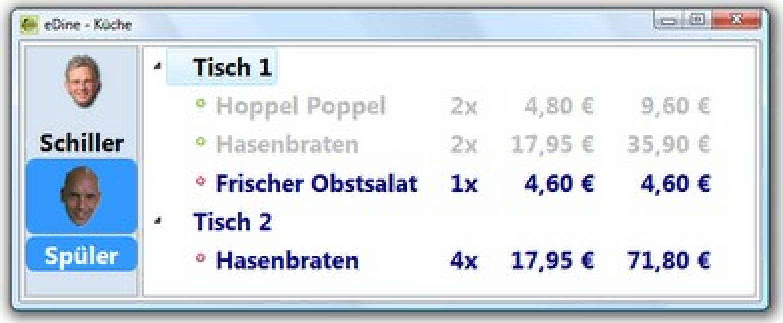
Connected application



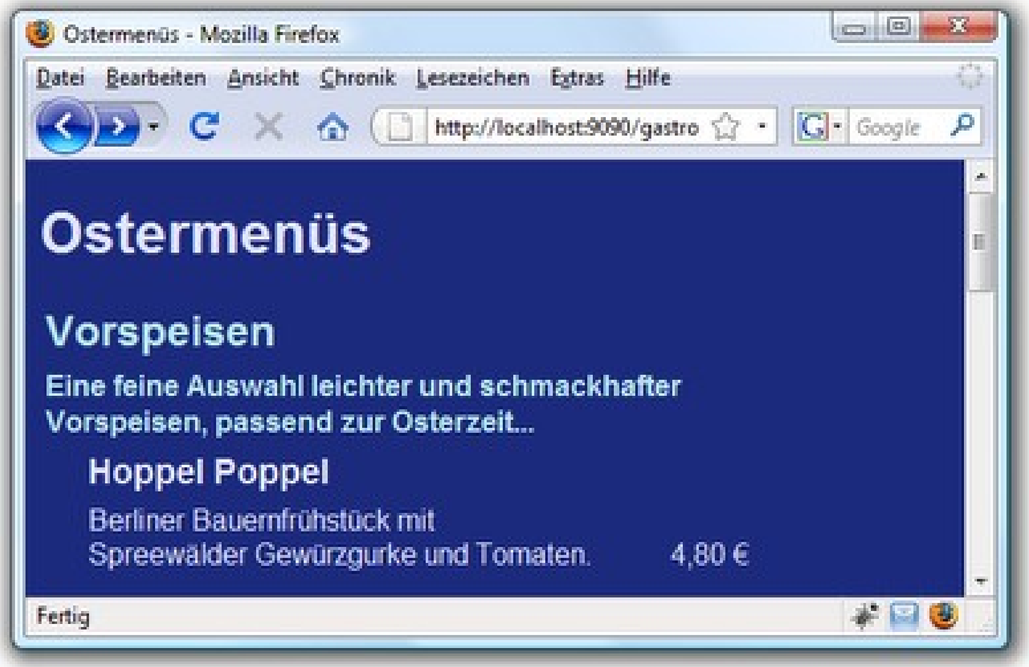
Hoppel Poppel

Berliner Bauernfrühstück mit Spreewälder Gewürzgurke und Tomaten.

4,80 €



Tisch 1			
Hoppel Poppel	2x	4,80 €	9,60 €
Hasenbraten	2x	17,95 €	35,90 €
Frischer Obstsalat	1x	4,60 €	4,60 €
Tisch 2			
Hasenbraten	4x	17,95 €	71,80 €



Ostermenüs

Vorspeisen

Eine feine Auswahl leichter und schmackhafter Vorspeisen, passend zur Osterzeit...

Hoppel Poppel

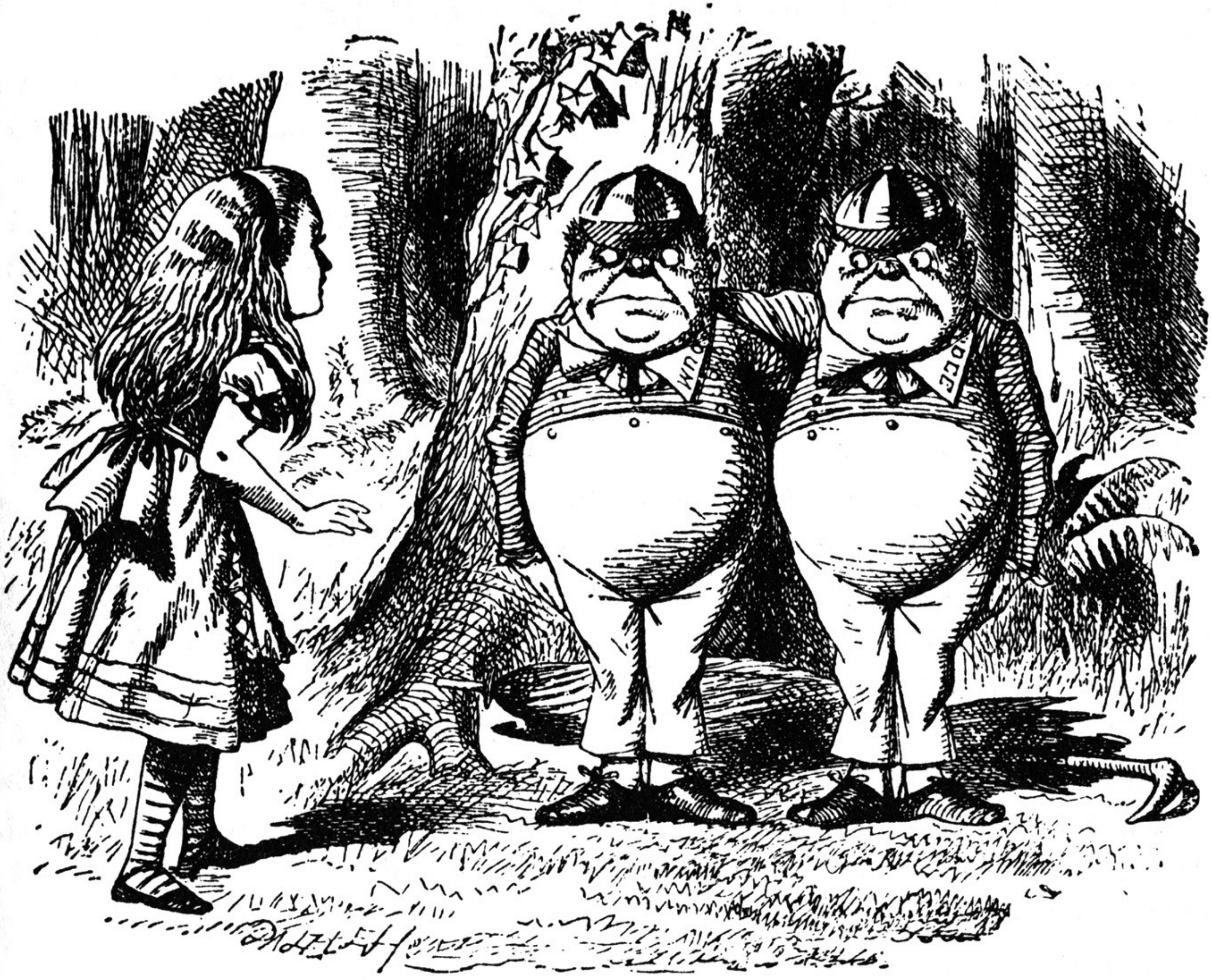
Berliner Bauernfrühstück mit Spreewälder Gewürzgurke und Tomaten. 4,80 €

You don't need
anything else,
do you ?

Two Faces

Technologies to build **applications**

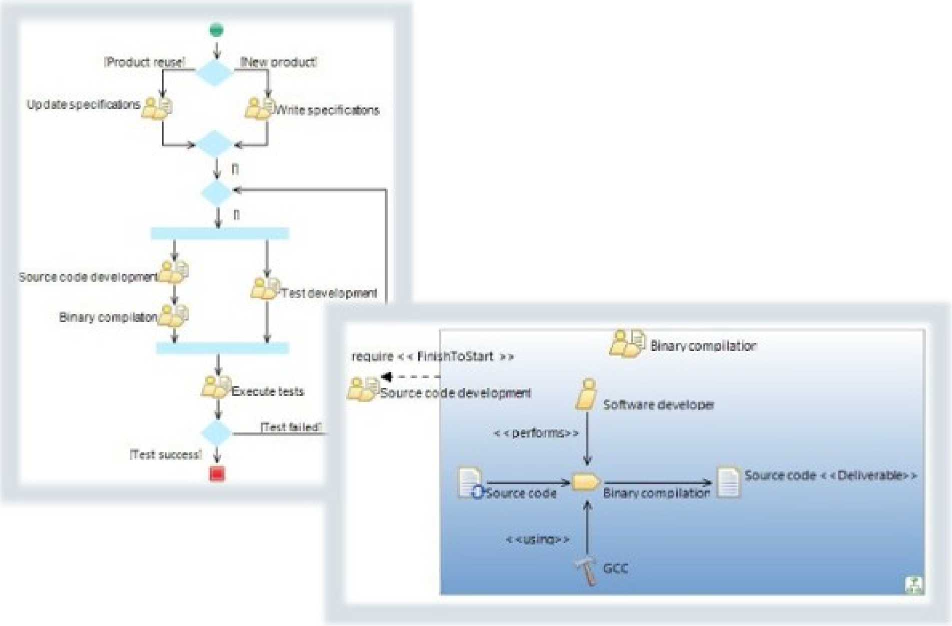
➡ Technologies to build **dedicated tools**

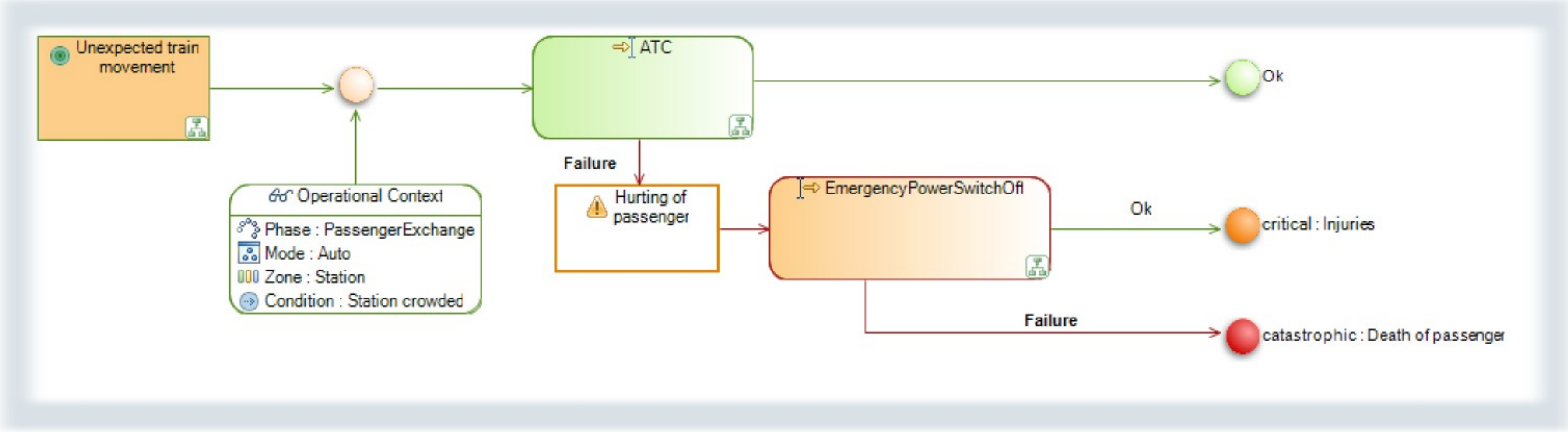


Building your own **Tools**

Graphical

- GMF has been around quite a long time already
- Graphiti is the new challenger

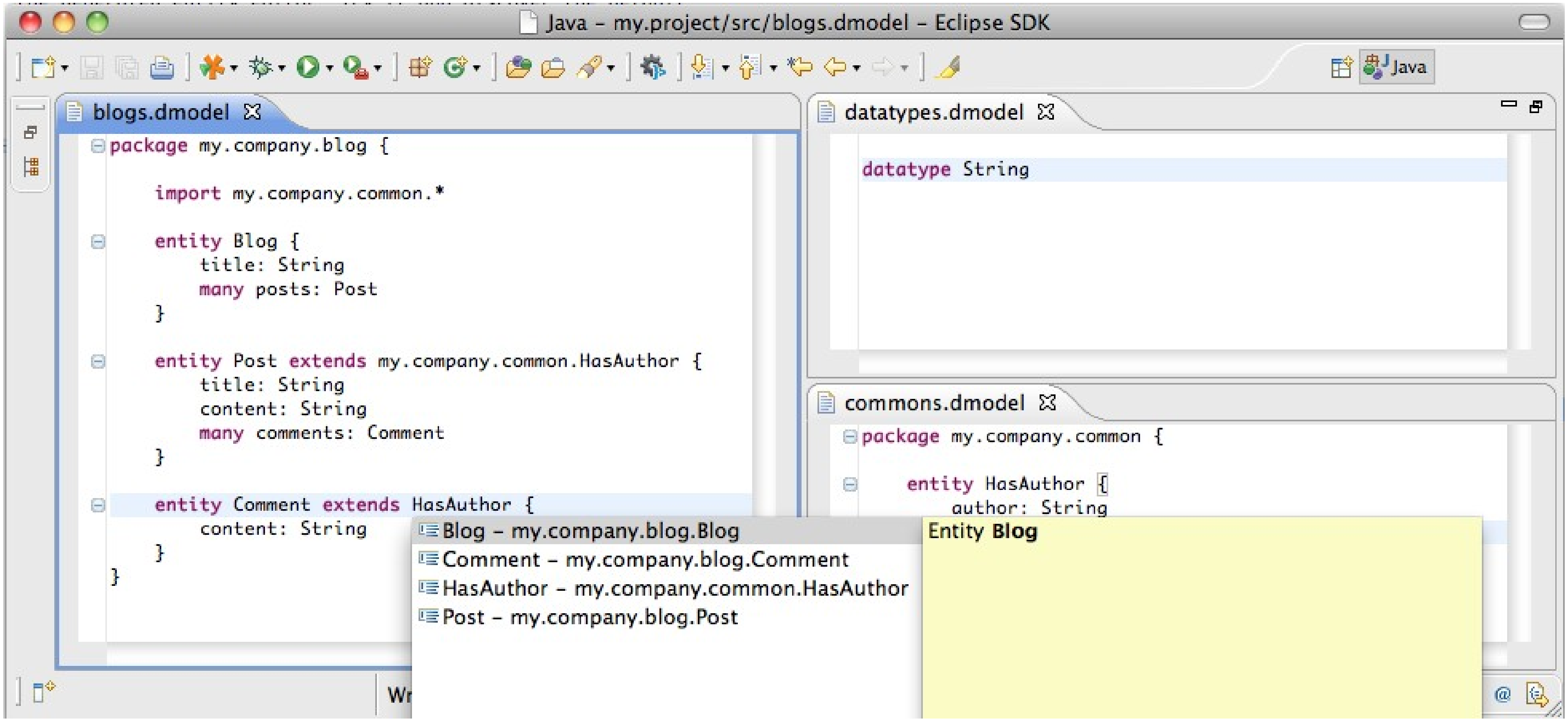




http://www.eclipse.org/modeling/gmp

Building your own **Tools**

Textual



```
package my.company.blog {

    import my.company.common.*

    entity Blog {
        title: String
        many posts: Post
    }

    entity Post extends my.company.common.HasAuthor {
        title: String
        content: String
        many comments: Comment
    }

    entity Comment extends HasAuthor {
        content: String
    }
}
```

```
datatype String
```

```
package my.company.common {

    entity HasAuthor {
        author: String
```

Blog – my.company.blog.Blog
Comment – my.company.blog.Comment
HasAuthor – my.company.common.HasAuthor
Post – my.company.blog.Post

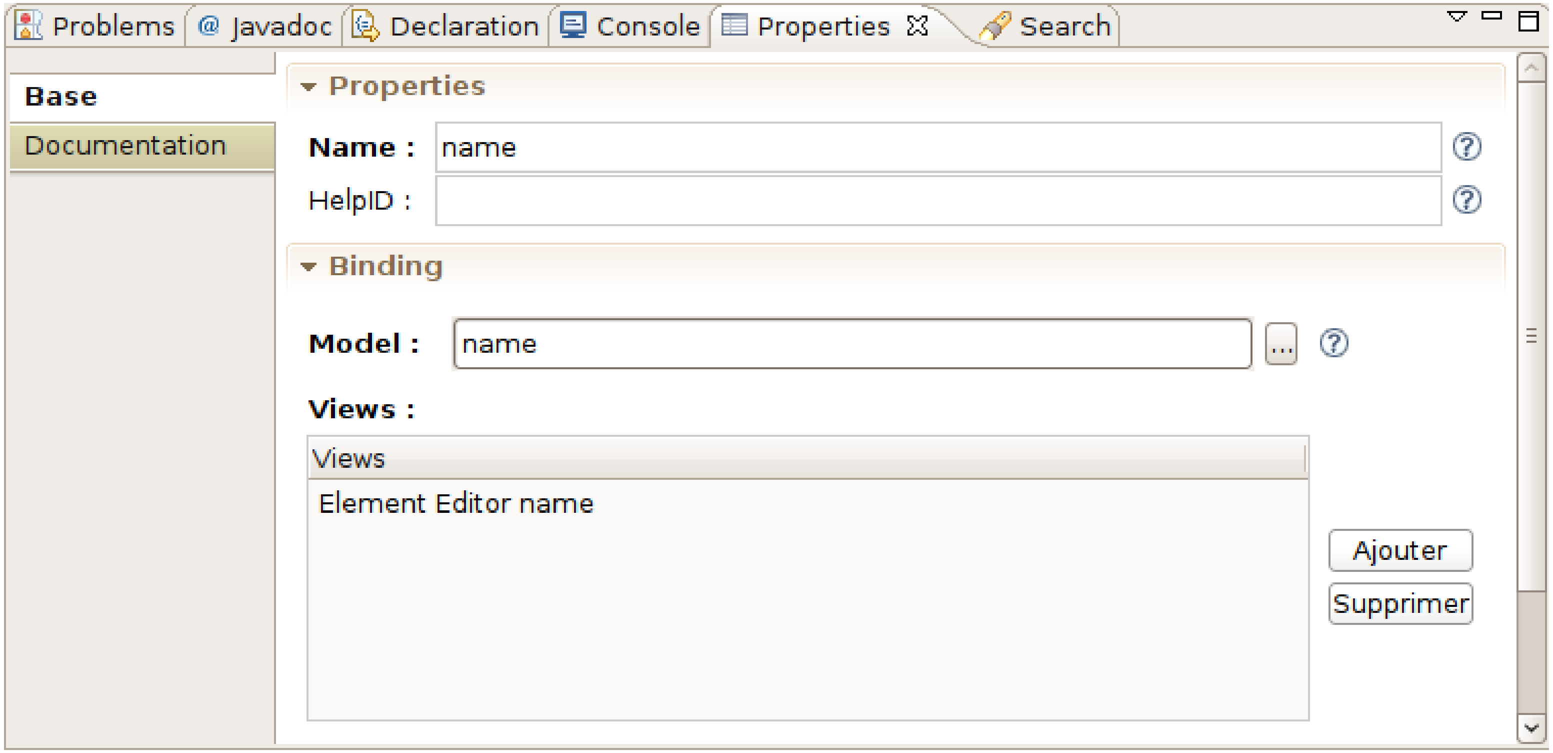
Entity **Blog**

http://www.eclipse.org/xtext

Building your own **Tools**

Forms

Extended Editing Framework aka **EEF**
(just graduated)



Name : name

HelpID :

Model : name

Views :

Views
Element Editor name

http://www.eclipse.org/eef

Building your own **Tools**

Automation

Template based generator

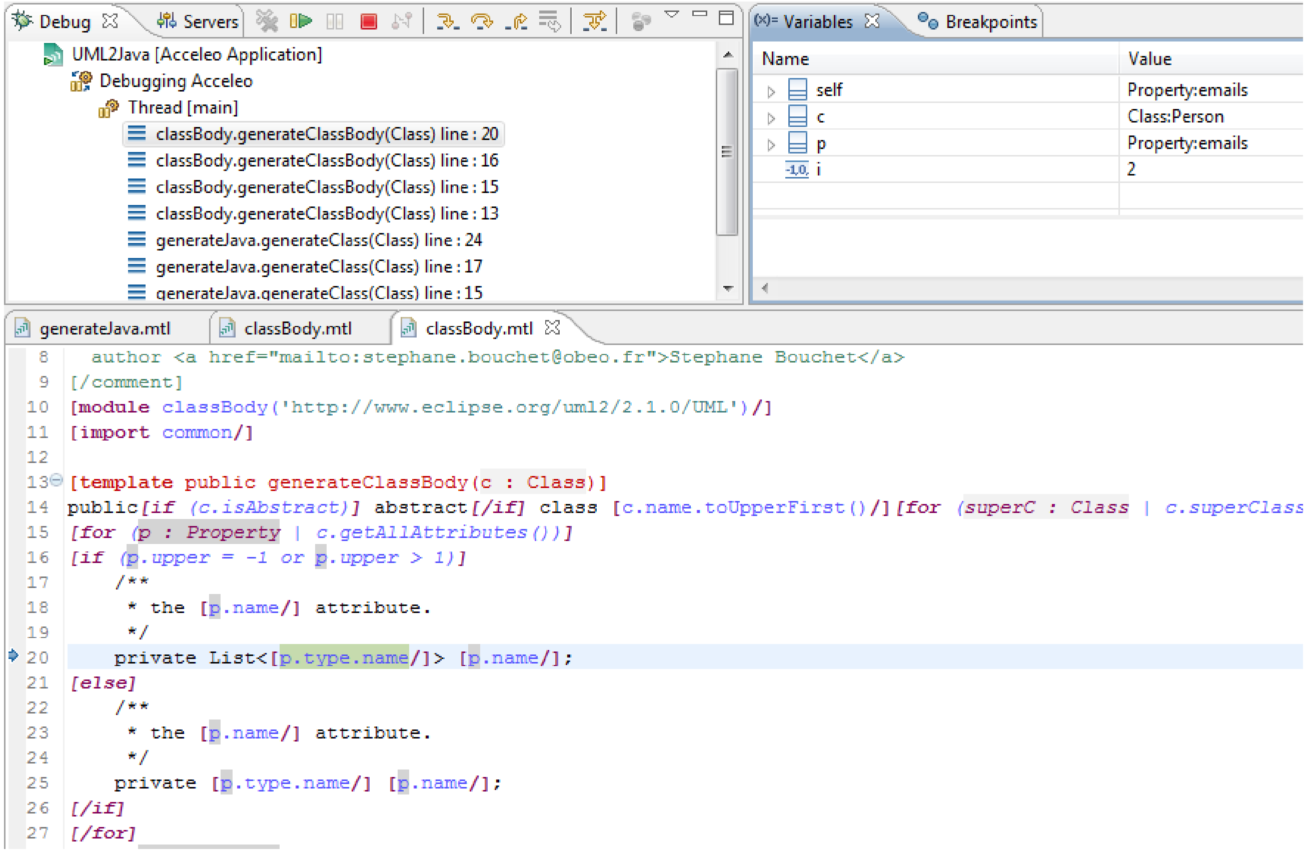
Complete **editor**

Debugger and **profiler**

Core runtime is **standalone**

Iterative generations

OMG std for generation



```
UML2Java [Acceleo Application]
  Debugging Acceleo
    Thread [main]
      classBody.generateClassBody(Class) line : 20
      classBody.generateClassBody(Class) line : 16
      classBody.generateClassBody(Class) line : 15
      classBody.generateClassBody(Class) line : 13
      generateJava.generateClass(Class) line : 24
      generateJava.generateClass(Class) line : 17
      generateJava.generateClass(Class) line : 15
```

Name	Value
self	Property:emails
c	Class:Person
p	Property:emails
i	2

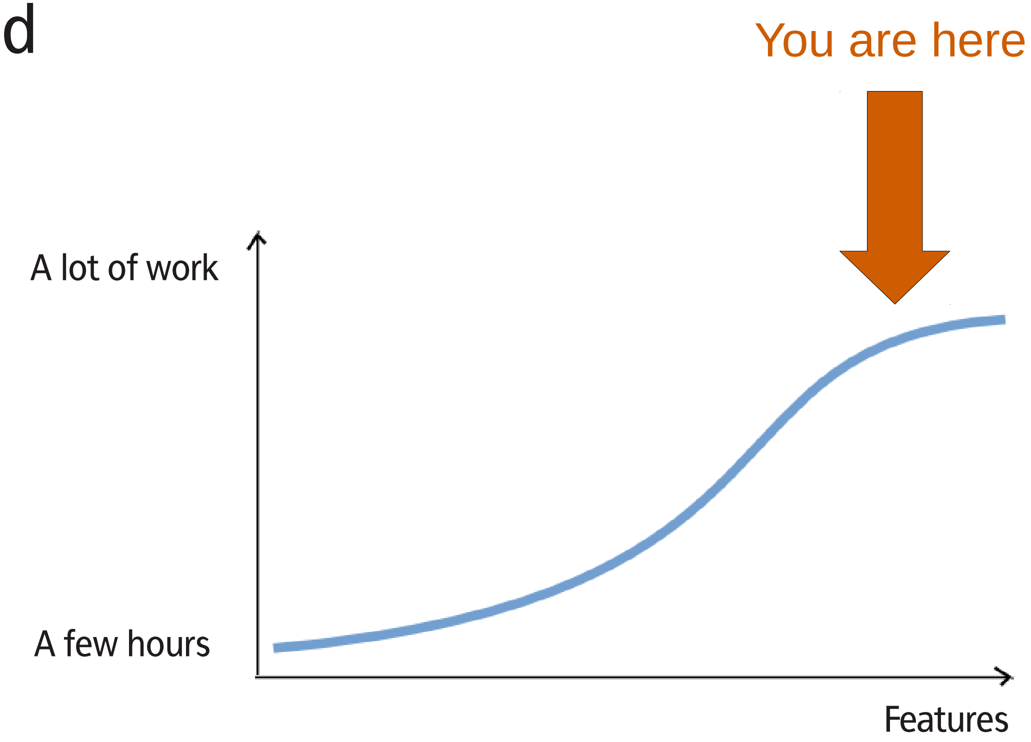
```
 8    author <a href="mailto:stephane.bouchet@obeo.fr">Stephane Bouchet</a>
 9  [/comment]
10  [module classBody('http://www.eclipse.org/uml2/2.1.0/UML')/]
11  [import common/]
12
13  [template public generateClassBody(c : Class)]
14  public[if (c.isAbstract)] abstract[/if] class [c.name.toUpperFirst()/][for (superC : Class | c.superClass
15  [for (p : Property | c.getAllAttributes())]
16  [if (p.upper = -1 or p.upper > 1)]
17      /**
18       * the [p.name/] attribute.
19       */
20      private List<[p.type.name/]> [p.name/];
21  [else]
22      /**
23       * the [p.name/] attribute.
24       */
25      private [p.type.name/] [p.name/];
26  [/if]
27  [/for]
```

Acceleo

http://www.eclipse.org/acceleo

a **Full blo**wn environment

- Large scale deployement and distribution
- Rich modeling tools
  (graphical, textual, forms)
- Fine grained collaborative support
- Ability to customize the tools and distribute the customizations
- Huge models scalability
- Customizable code generators
- Tests on all this tooling

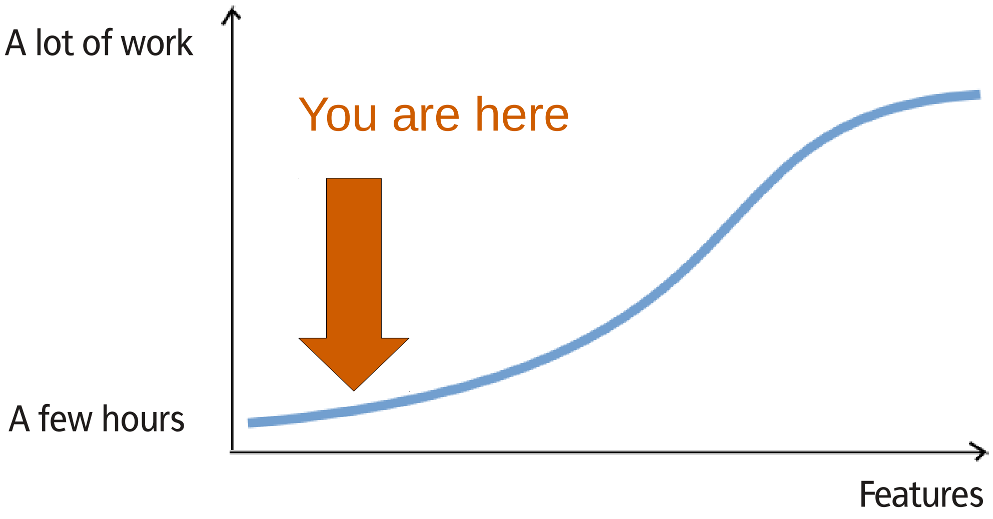


easy **Automation**

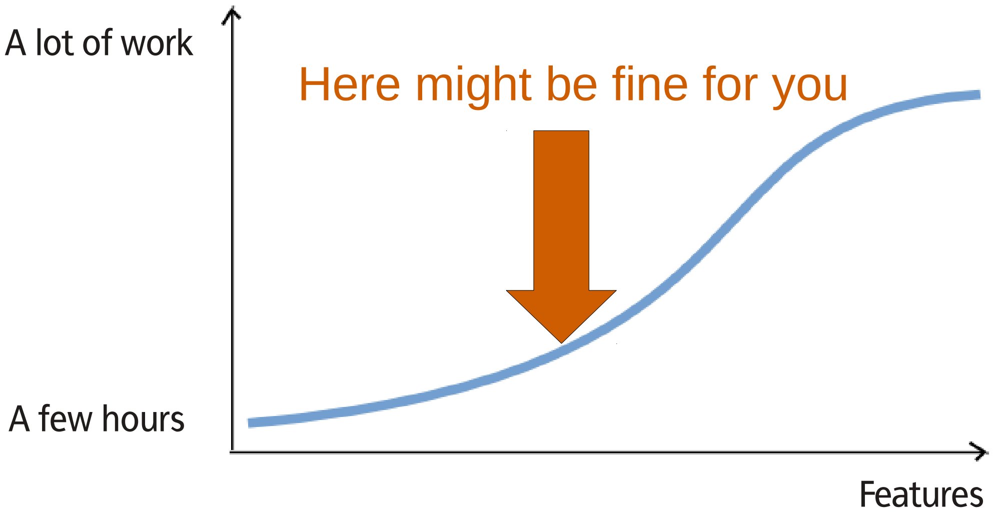
- Basic Editing capabilities

  Reflective Editors
- Team-wide deploiement through SCM
- Classic collaboration processes

  EMF compare
- **Code generation!**



Every intermediate level is valid too!



Let's **Try** ...

You need (*)

- Eclipse
- EMF
- Acceleo

We will build:

A model to keep track of our app features among n versions

(*) Two clicks to install from the Eclipse Marketplace

Is that a **Good** model?

1. Its answering a question

   What features have been updated/created in version 1.234?

2. It is used

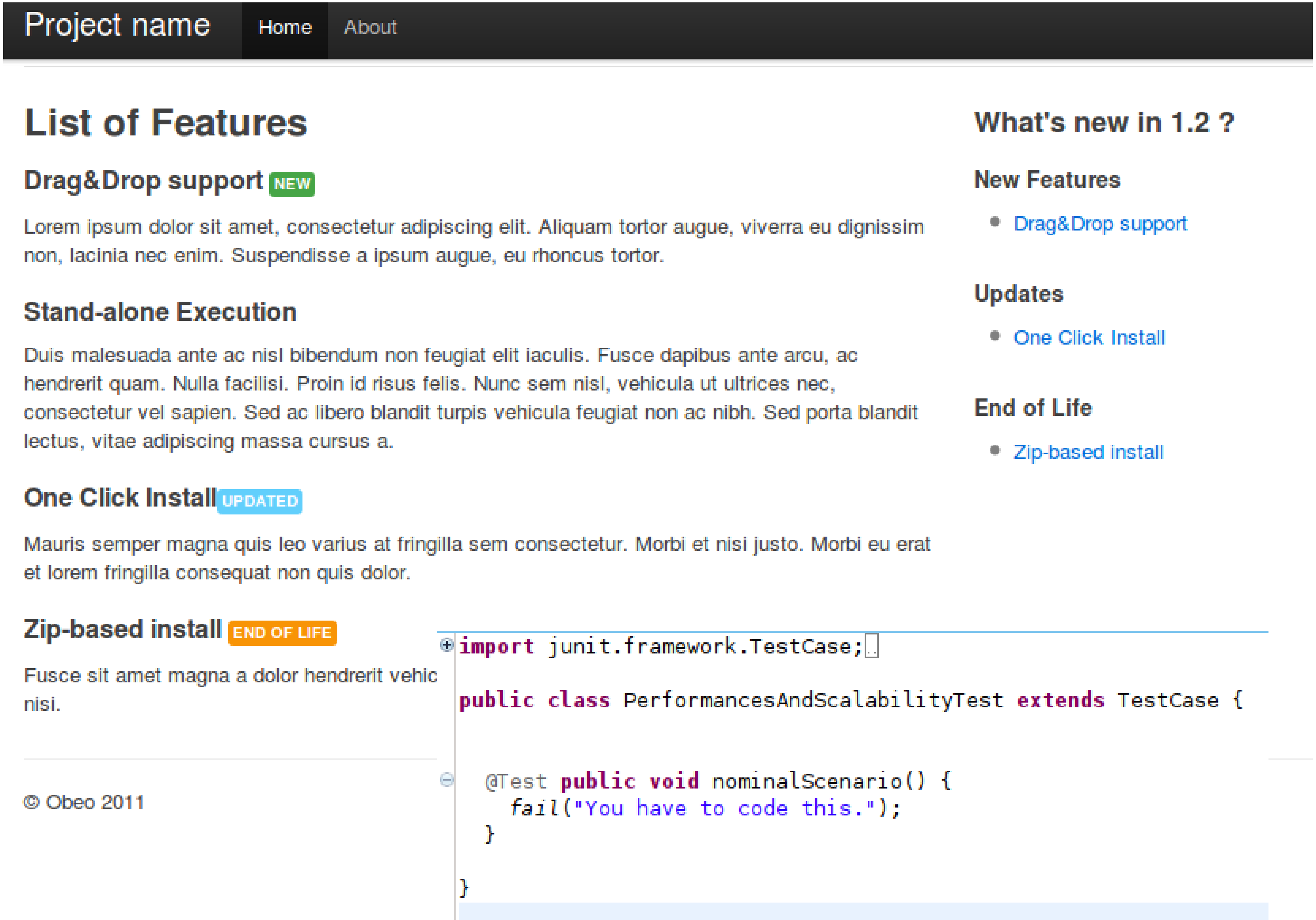
   to generate skeletons of tests
   to generate documentation

how to **Proceed**?

1. code a **prototype** of what you expect
2. create the **domain model**
3. create an **instance** of this domain model
4. create a generation **template**
5. **launch**

how to **Proceed**?

1. code a **prototype** of what you expect



List of Features

Drag&Drop support NEW

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam tortor augue, viverra eu dignissim non, lacinia nec enim. Suspendisse a ipsum augue, eu rhoncus tortor.

Stand-alone Execution

Duis malesuada ante ac nisl bibendum non feugiat elit iaculis. Fusce dapibus ante arcu, ac hendrerit quam. Nulla facilisi. Proin id risus felis. Nunc sem nisl, vehicula ut ultrices nec, consectetur vel sapien. Sed ac libero blandit turpis vehicula feugiat non ac nibh. Sed porta blandit lectus, vitae adipiscing massa cursus a.

One Click Install UPDATED

Mauris semper magna quis leo varius at fringilla sem consectetur. Morbi et nisi justo. Morbi eu erat et lorem fringilla consequat non quis dolor.

Zip-based install END OF LIFE

Fusce sit amet magna a dolor hendrerit vehic
nisi.

© Obeo 2011

What's new in 1.2 ?

New Features

- Drag&Drop support

Updates

- One Click Install

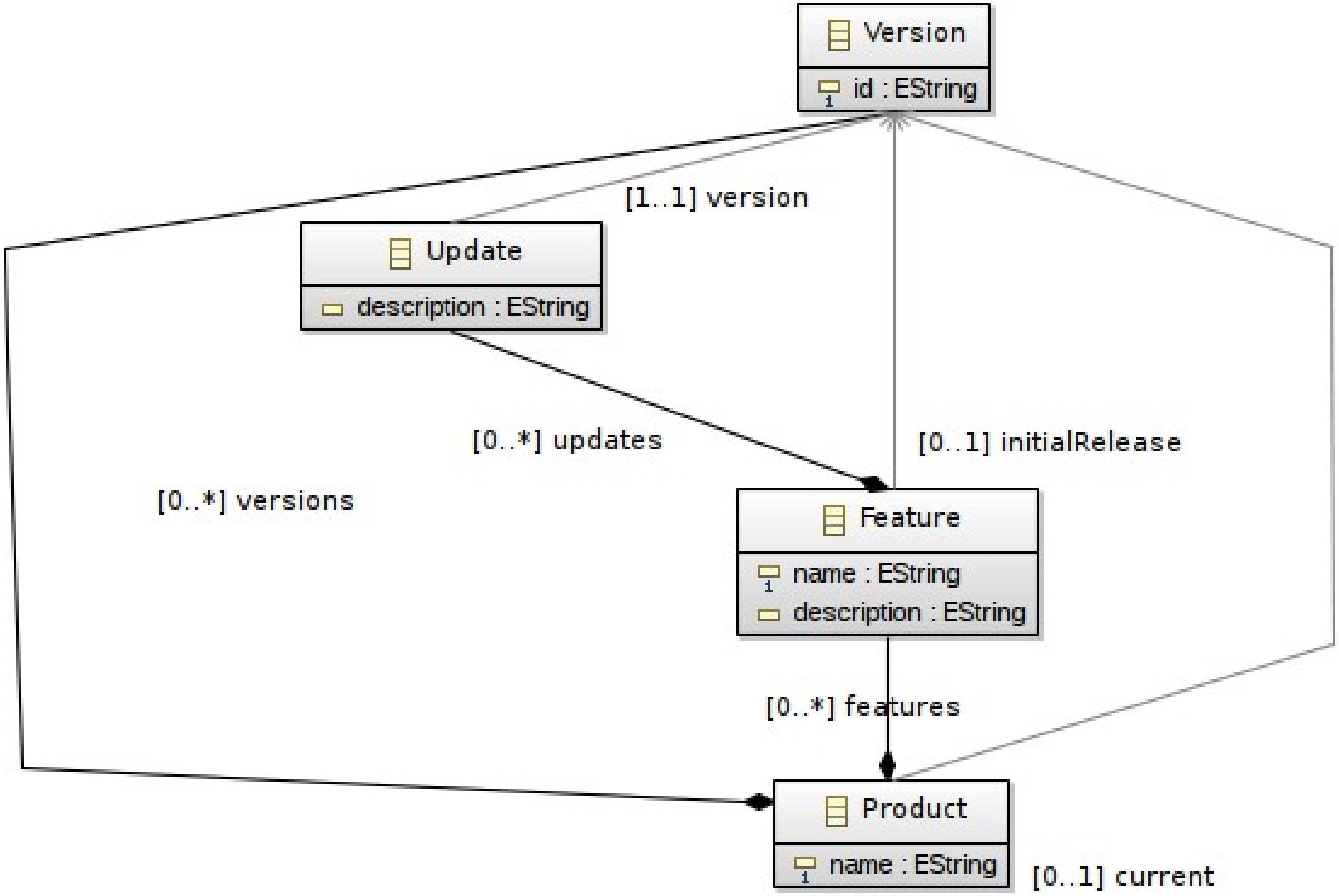
End of Life

- Zip-based install

```
import junit.framework.TestCase;

public class PerformancesAndScalabilityTest extends TestCase {


  @Test public void nominalScenario() {
    fail("You have to code this.");
  }

}
```

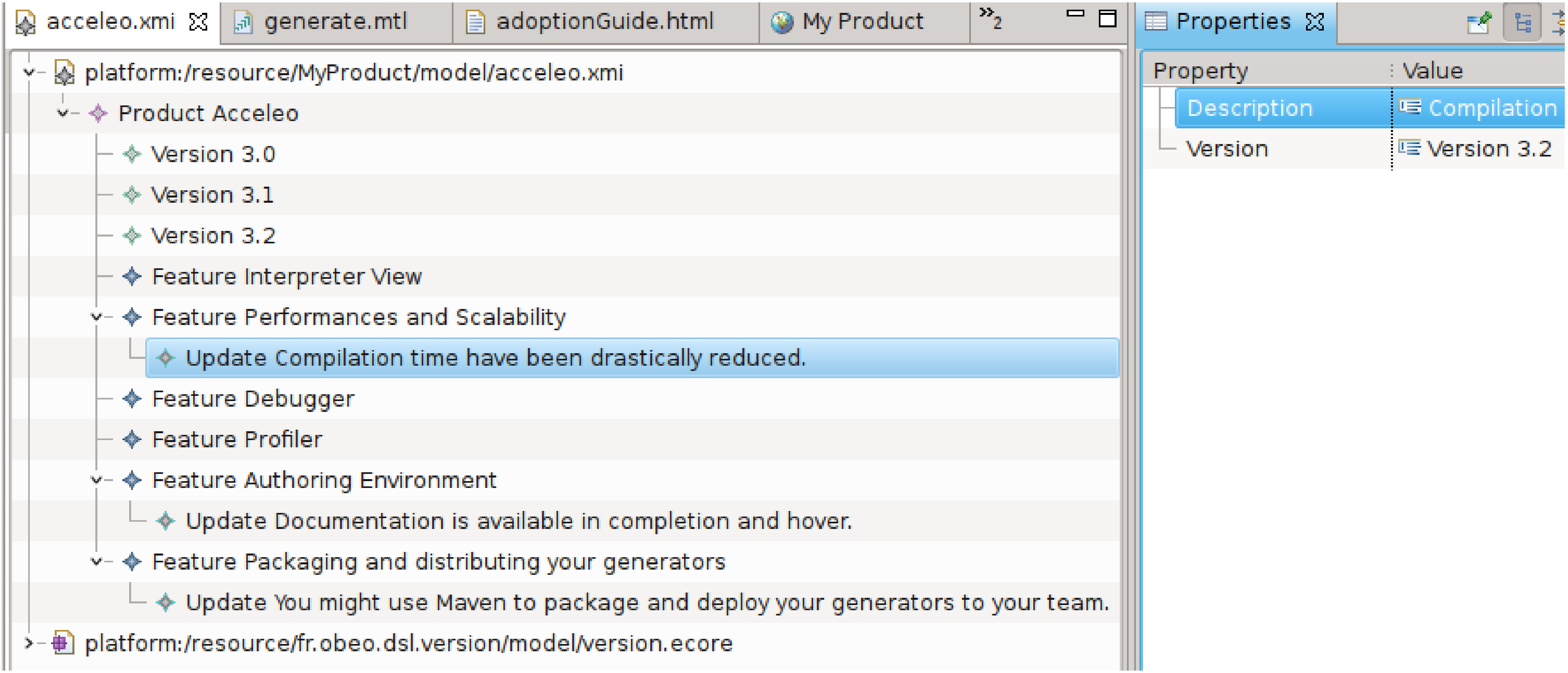
how to **Proceed**?

2. create the **domain model**



how to **Proceed**?

3. create an **instance** of this domain model

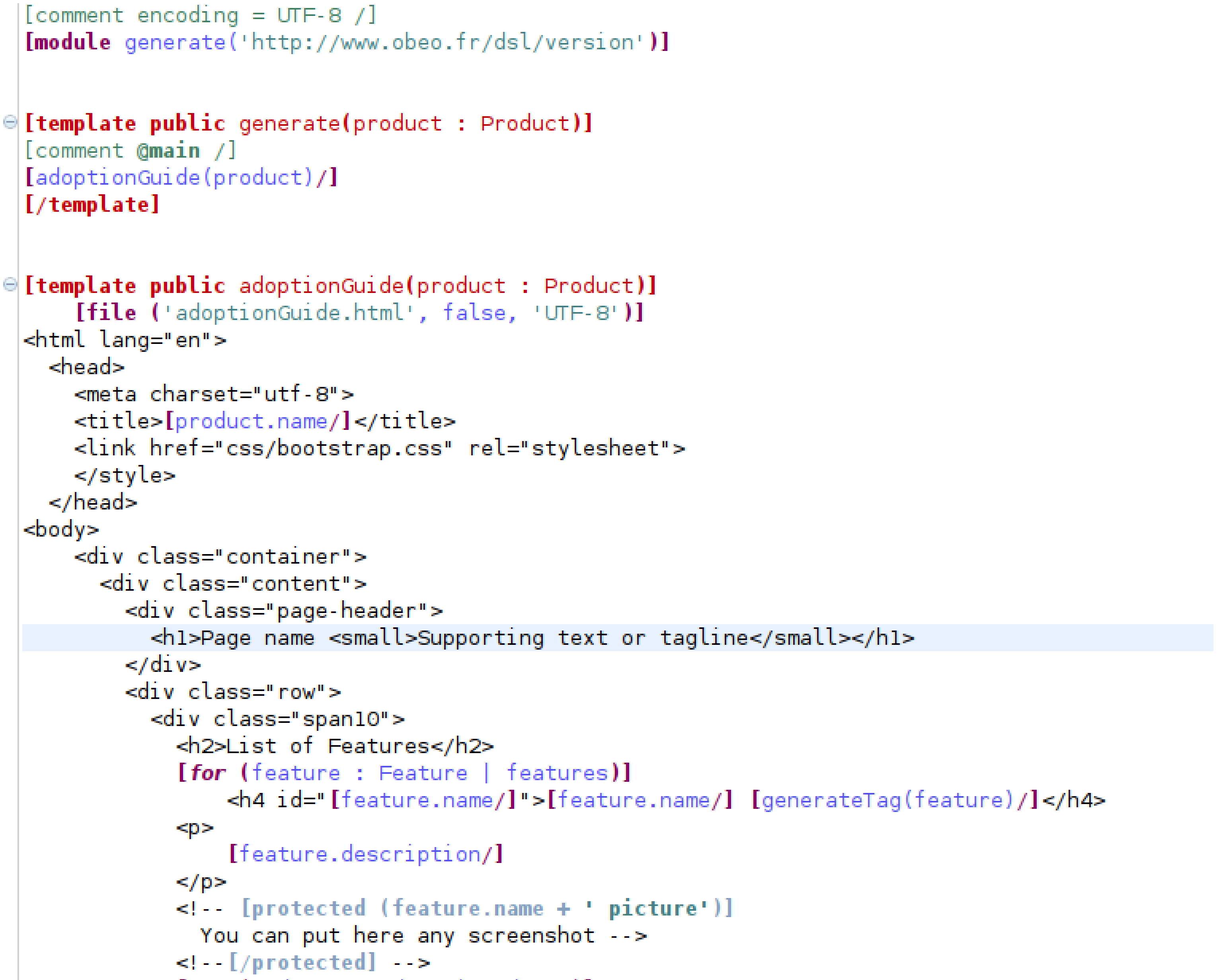


- platform:/resource/MyProduct/model/acceleo.xmi
  - Product Acceleo
    - Version 3.0
    - Version 3.1
    - Version 3.2
    - Feature Interpreter View
    - Feature Performances and Scalability
      - Update Compilation time have been drastically reduced.
    - Feature Debugger
    - Feature Profiler
    - Feature Authoring Environment
      - Update Documentation is available in completion and hover.
    - Feature Packaging and distributing your generators
      - Update You might use Maven to package and deploy your generators to your team.
- platform:/resource/fr.obeo.dsl.version/model/version.ecore

Property	Value
Description	Compilation
Version	Version 3.2

how to **Proceed**?

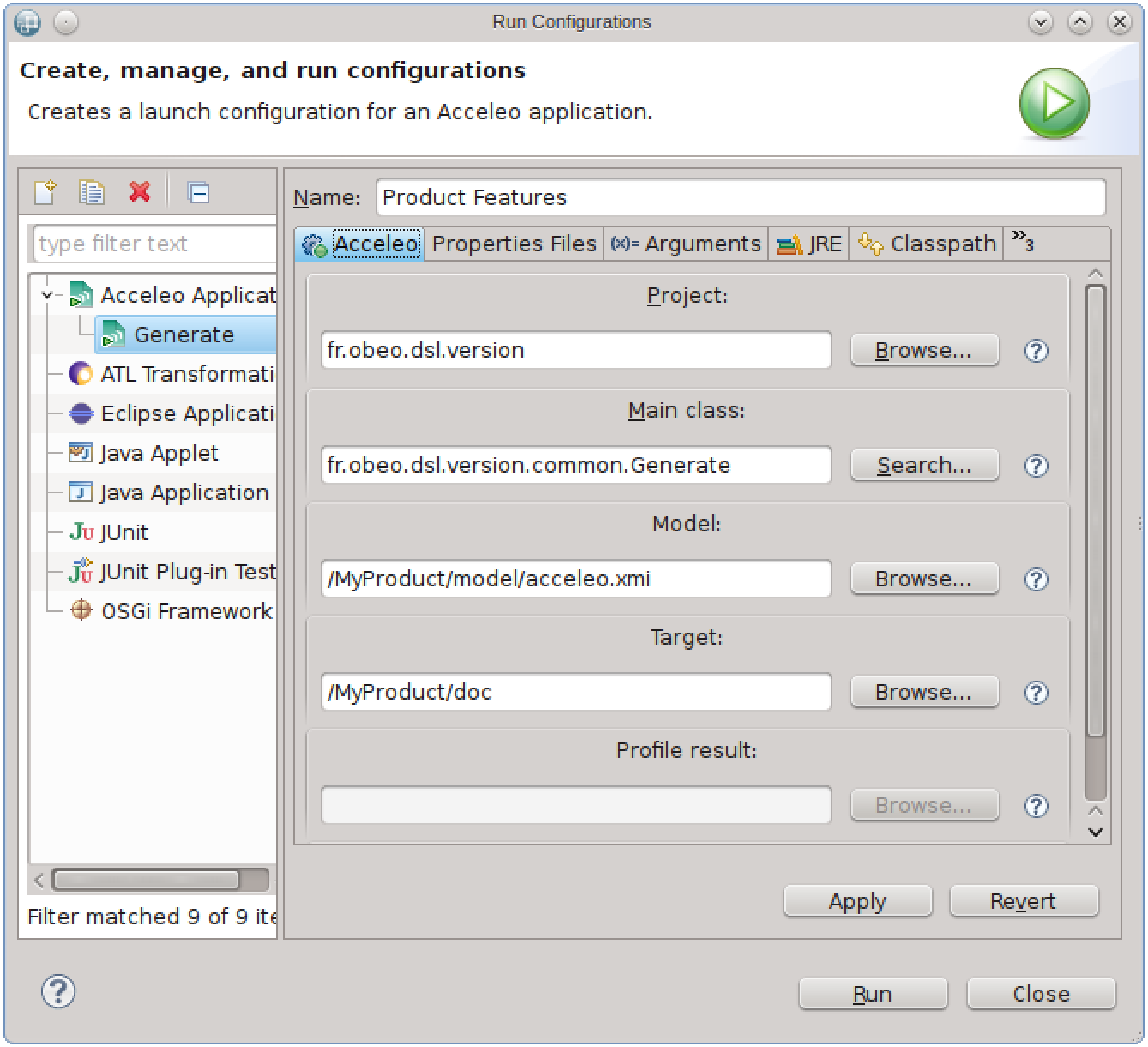
4. create a generation **template**



```
[comment encoding = UTF-8 /]
[module generate('http://www.obeo.fr/dsl/version')]


[template public generate(product : Product)]
[comment @main /]
[adoptionGuide(product)/]
[/template]


[template public adoptionGuide(product : Product)]
  [file ('adoptionGuide.html', false, 'UTF-8')]
<html lang="en">
 <head>
  <meta charset="utf-8">
  <title>[product.name/]</title>
  <link href="css/bootstrap.css" rel="stylesheet">
  </style>
 </head>
<body>
  <div class="container">
   <div class="content">
    <div class="page-header">
     <h1>Page name <small>Supporting text or tagline</small></h1>
    </div>
    <div class="row">
     <div class="span10">
      <h2>List of Features</h2>
      [for (feature : Feature | features)]
        <h4 id="[feature.name/]">[feature.name/] [generateTag(feature)/]</h4>
      <p>
        [feature.description/]
      </p>
      <!-- [protected (feature.name + ' picture')]
       You can put here any screenshot -->
      <!--[/protected] -->
```

how to **Proceed**?

5. **launch**



Run Configurations

Create, manage, and run configurations

Creates a launch configuration for an Acceleo application.

Name: Product Features

Project: fr.obeo.dsl.version

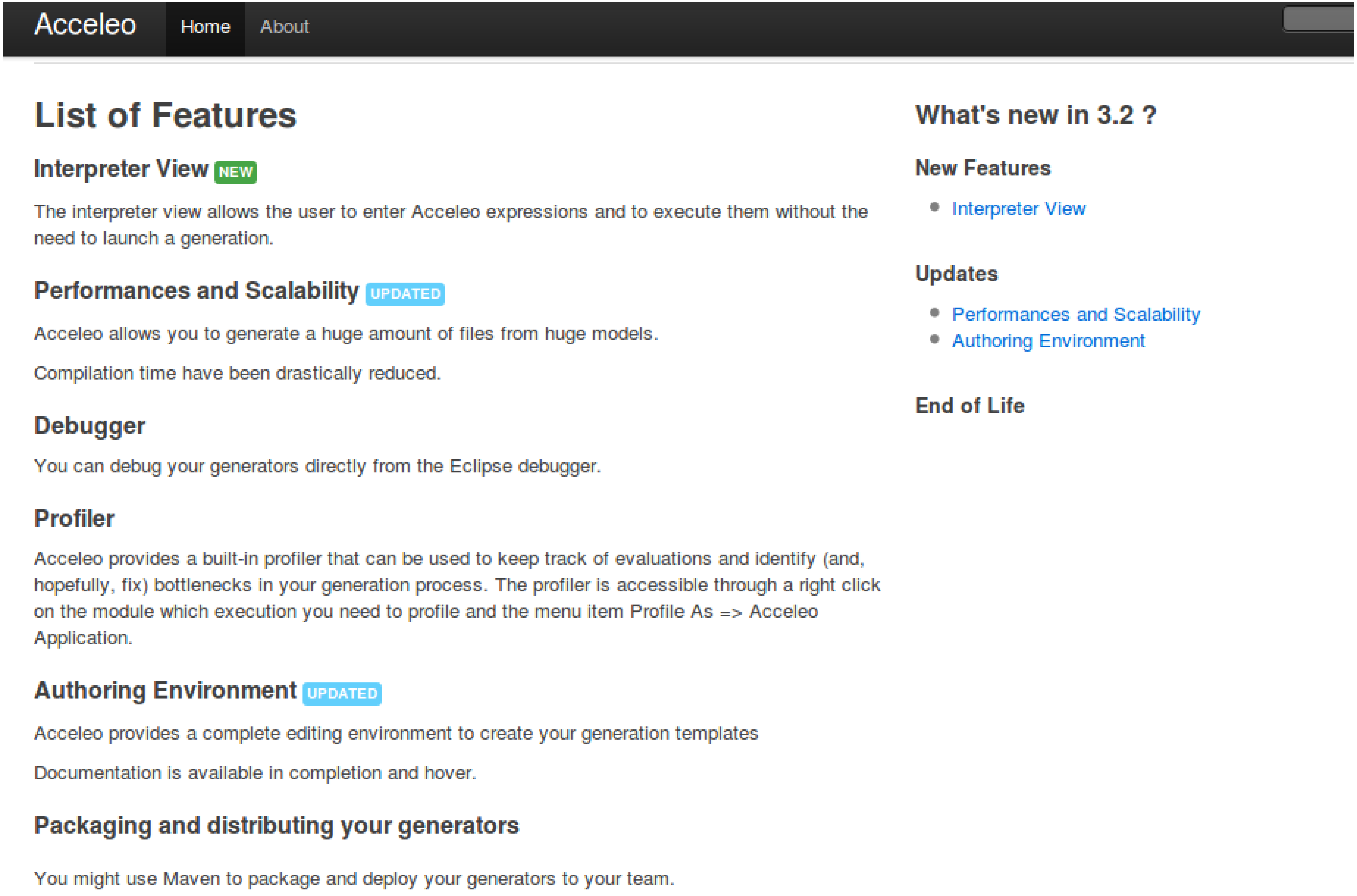
Main class: fr.obeo.dsl.version.common.Generate

Model: /MyProduct/model/acceleo.xmi

Target: /MyProduct/doc

Profile result:

Result?



List of Features

Interpreter View NEW

The interpreter view allows the user to enter Acceleo expressions and to execute them without the need to launch a generation.

Performances and Scalability UPDATED

Acceleo allows you to generate a huge amount of files from huge models.

Compilation time have been drastically reduced.

Debugger

You can debug your generators directly from the Eclipse debugger.

Profiler

Acceleo provides a built-in profiler that can be used to keep track of evaluations and identify (and, hopefully, fix) bottlenecks in your generation process. The profiler is accessible through a right click on the module which execution you need to profile and the menu item Profile As => Acceleo Application.

Authoring Environment UPDATED

Acceleo provides a complete editing environment to create your generation templates

Documentation is available in completion and hover.

Packaging and distributing your generators

You might use Maven to package and deploy your generators to your team.

What's new in 3.2 ?

New Features

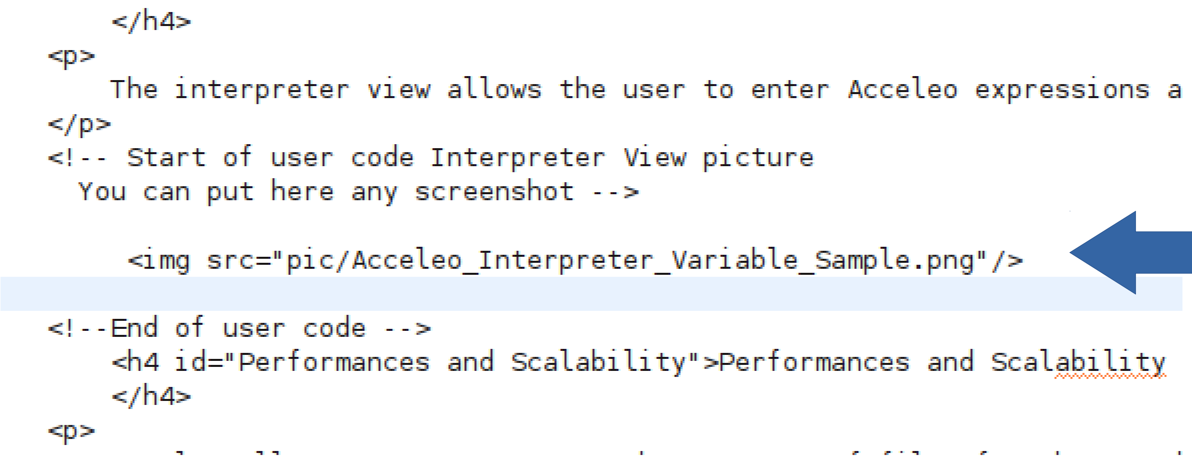
- Interpreter View

Updates

- Performances and Scalability
- Authoring Environment

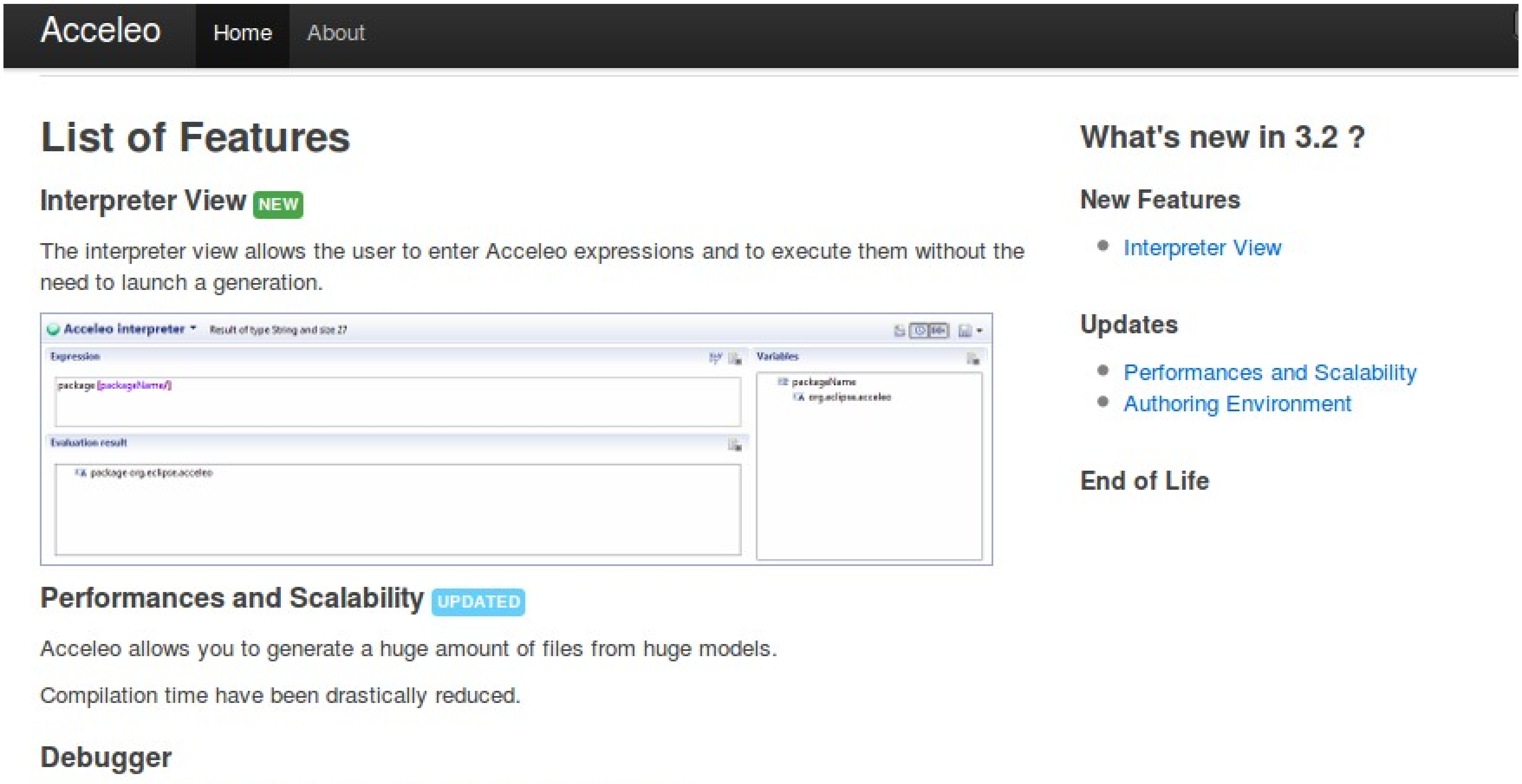
End of Life

Customize it



```
    </h4>
<p>
    The interpreter view allows the user to enter Acceleo expressions a
</p>
<!-- Start of user code Interpreter View picture
  You can put here any screenshot -->

     <img src="pic/Acceleo_Interpreter_Variable_Sample.png"/>

<!--End of user code -->
    <h4 id="Performances and Scalability">Performances and Scalability
    </h4>
<p>
```

Adding non-generated doc



Acceleo   Home   About

List of Features

Interpreter View NEW

The interpreter view allows the user to enter Acceleo expressions and to execute them without the need to launch a generation.

Performances and Scalability UPDATED

Acceleo allows you to generate a huge amount of files from huge models.

Compilation time have been drastically reduced.

Debugger

What's new in 3.2 ?

New Features

- Interpreter View

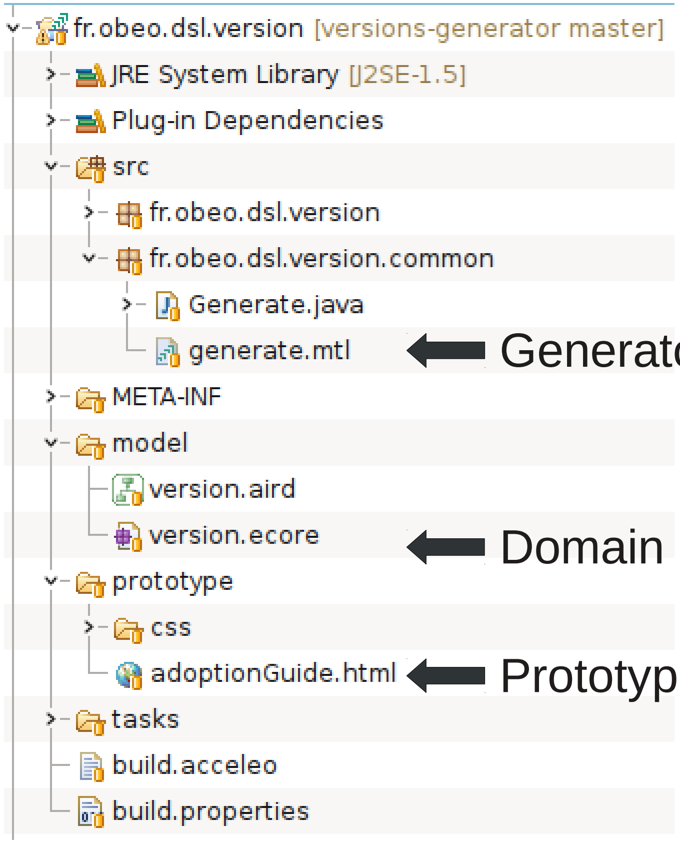
Updates

- Performances and Scalability
- Authoring Environment

End of Life

What are the **Benefits?**

We have a **consistent** vision of our features among versions
We produce a useful **deliverable** for our end users
Our tooling is versioned with its «project» **dedicated branch**
Anybody from the team can maintain and change the tooling
It took **2 hours** to set up

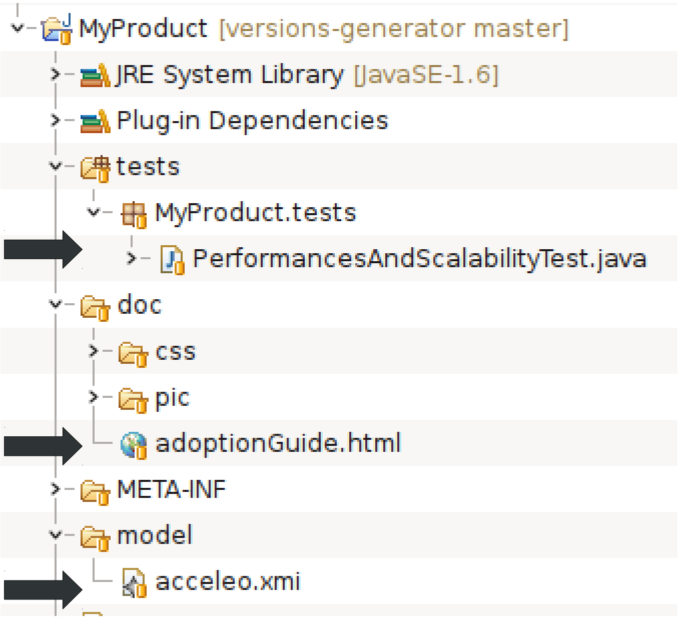


```
fr.obeo.dsl.version [versions-generator master]
  JRE System Library [J2SE-1.5]
  Plug-in Dependencies
  src
    fr.obeo.dsl.version
    fr.obeo.dsl.version.common
      Generate.java
      generate.mtl
  META-INF
  model
    version.aird
    version.ecore
  prototype
    css
    adoptionGuide.html
  tasks
  build.acceleo
  build.properties
```

Generator

Domain model

Prototype



```
MyProduct [versions-generator master]
  JRE System Library [JavaSE-1.6]
  Plug-in Dependencies
  tests
    MyProduct.tests
      PerformancesAndScalabilityTest.java
  doc
    css
    pic
    adoptionGuide.html
  META-INF
  model
    acceleo.xmi
```

Generated tests

Generated doc

Model instance

https://github.com/cbrun/versions-generator

Rich and **diverse** Ecosystem

You can build
- **applications**
- dedicated **tools**

You can try at a very **low cost**

Try it !

References

The Eclipse Modeling Package
http://www.eclipse.org/modeling/amalgam/

The Eclipse Forum
http://www.eclipse.org/forums/

The Eclipse Modeling Cross Projects Example
http://goo.gl/51RuK

The versions generator example:
https://github.com/cbrun/versions-generator

Obeo Network
http://www.obeonetwork.org/

Me
@bruncedric on twitter
http://model-driven-blogging.blogspot.com/

« Alice in Wonderland » pictures are from John Tenniel in public domain since 1964