

Modelos de Software

Ingeniería en Sistemas de Información
2018

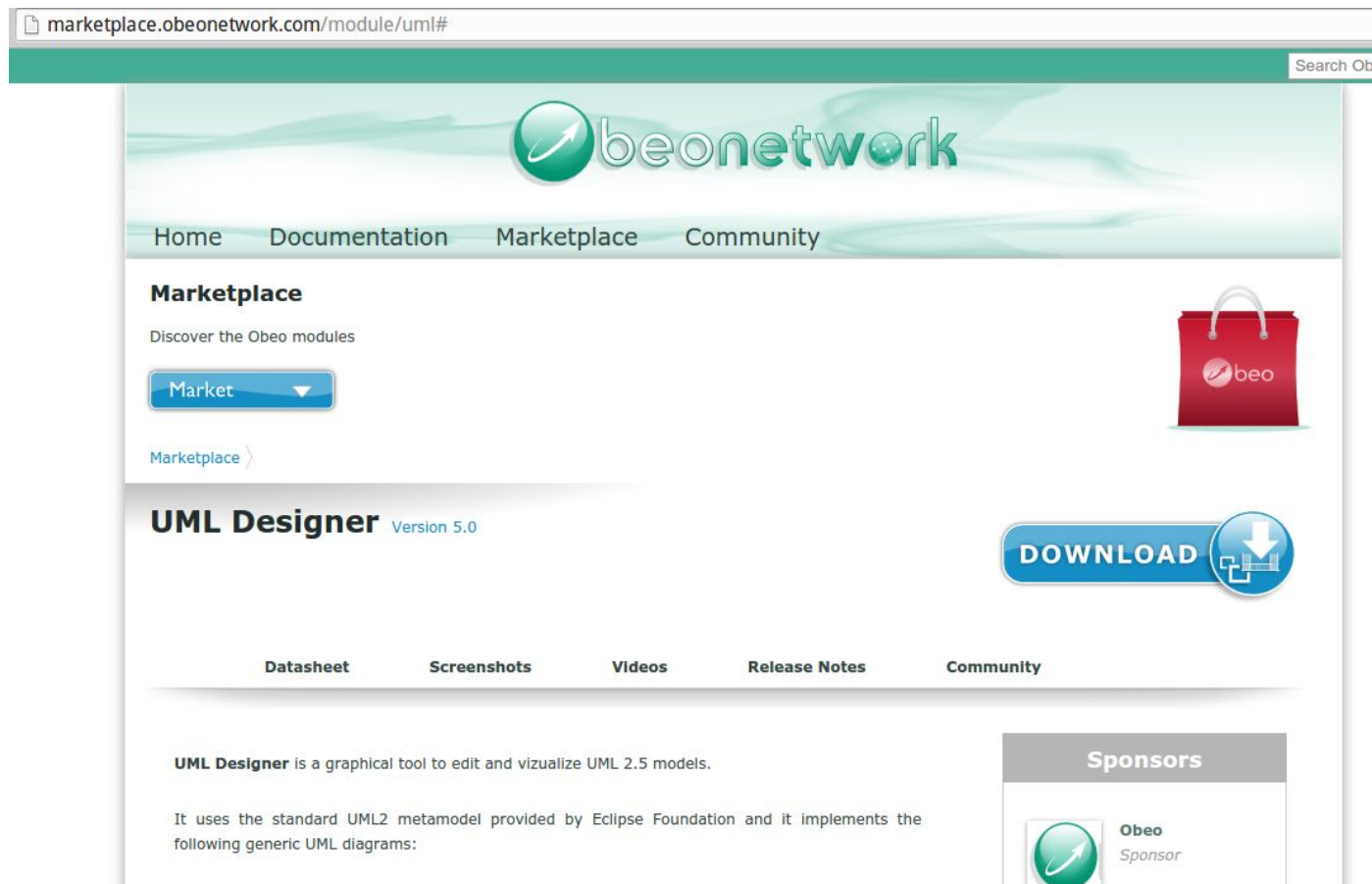
Modelos de Software

UML

La línea que separa el modelado de la programación a veces se vuelve algo difusa. Existen muchas herramientas que permiten transformar un diagrama de clases UML a código (y viceversa).

Modelos de Software

UML to Java



The screenshot shows a web browser window with the URL `marketplace.obeonetwork.com/module/uml#`. The page features the Obeo Network logo at the top, a navigation menu with links for Home, Documentation, Marketplace, and Community, and a search bar. The main content area is titled "Marketplace" and includes a "Discover the Obeo modules" section with a "Market" dropdown menu and a red shopping bag icon. Below this, the "UML Designer Version 5.0" product is highlighted with a prominent "DOWNLOAD" button. A secondary navigation bar offers links for Datasheet, Screenshots, Videos, Release Notes, and Community. The page also contains a descriptive paragraph about UML Designer and a "Sponsors" section featuring the Obeo Sponsor logo.

marketplace.obeonetwork.com/module/uml#

Search Ob

Home Documentation Marketplace Community

Marketplace

Discover the Obeo modules

Market

Marketplace >

UML Designer Version 5.0

DOWNLOAD

Datasheet Screenshots Videos Release Notes Community

UML Designer is a graphical tool to edit and visualize UML 2.5 models.

It uses the standard UML2 metamodel provided by Eclipse Foundation and it implements the following generic UML diagrams:

Sponsors

Obeo Sponsor

Modelos de Software

UML to Java

Uml to Java Generator Version 3.0.0

DOWNLOAD



[Datasheet](#)

[Screenshots](#)

[Release Notes](#)

[Community](#)

This generator allows you to produce Java source code from a UML model. This module uses the standard UML2 metamodel provided by Eclipse Foundation and as such is compatible with any UML tools using this standard like [UML Designer](#). It is based on the award winning code generation project, [Acceleo](#).

Created by the members of the Acceleo development team, this generator is part of the [Eclipse UML Generators](#) project. It is highly integrated in Eclipse with a dedicated launch configuration to keep track and your generations. This generator also includes a builder that can be activated to monitor your project and generate your code if your UML model is modified, keeping your code synchronized with your model. This module also contains an integrated documentation

Sponsors



Obeo
Sponsor

The Team

Modelos de Software

UML

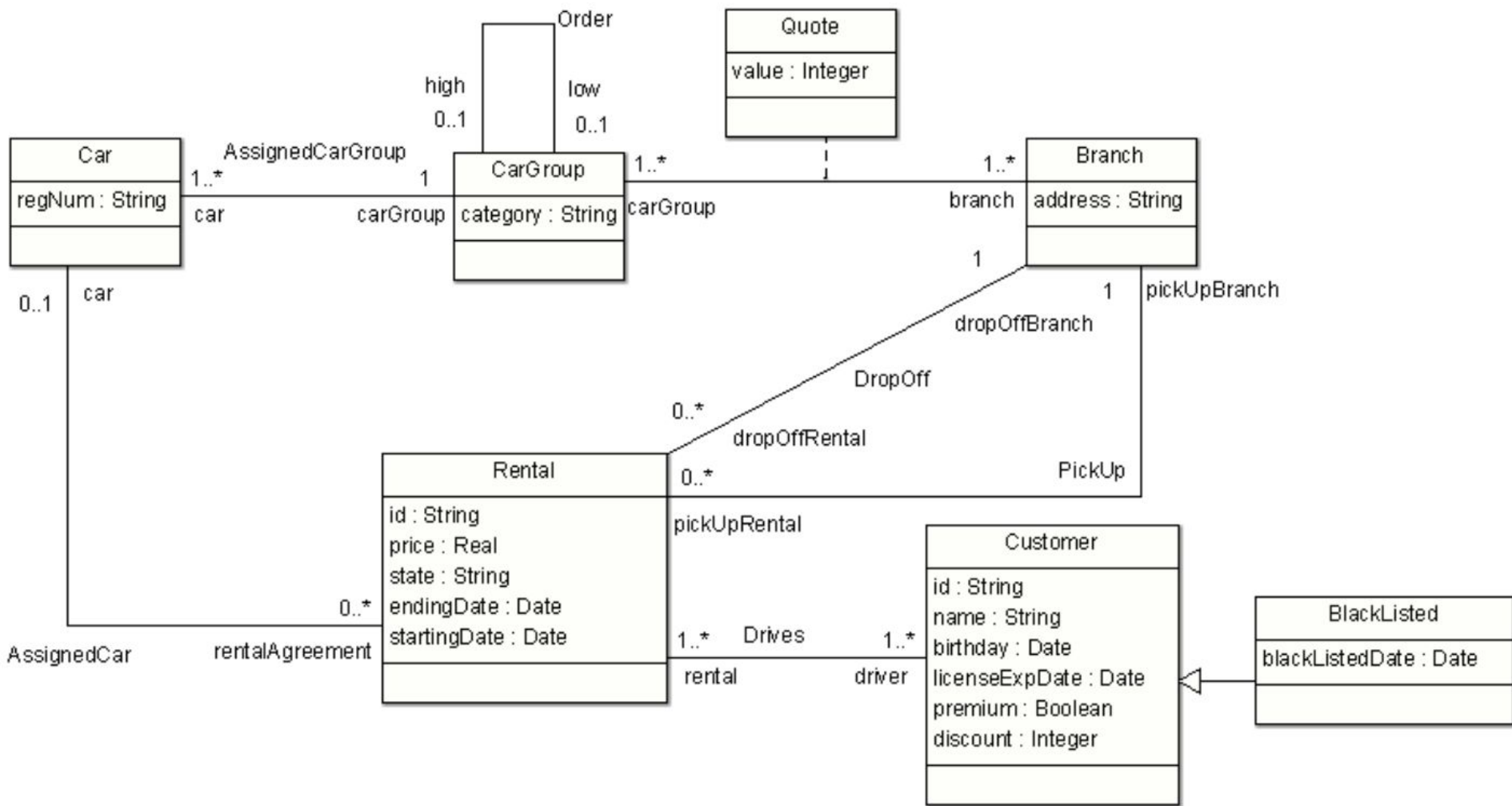
¿Qué otras cosas se pueden hacer con UML?

Modelos de Software

Limitaciones de UML

Para que un lenguaje de modelado sea claro y legible se debe limitar su expresividad. Esto significa que una representación gráfica, como es el diagrama de clase, sólo presenta un subconjunto de todo lo que es importante.

Modelos de Software



Modelos de Software

UML

Aunque este diagrama parece completo, hay cuestiones del sistema que no se están reflejando y son propias del problema:

¿Cómo se calcula el precio de un alquiler?

¿Puede una persona en una black list volver a alquilar?

¿Los lugares de pick up y drop off pueden ser diferentes?

Modelos de Software

UML

Existen lenguajes que permiten introducir expresiones que complementan el modelado. Escribir notas en castellano junto al diagrama es una forma de complementar. Hoy vamos a ver metodologías más formales.

Modelos de Software

Object Constraint Language (OCL)

Modelos de Software

UML + OCL

OCL es un lenguaje de especificación tipificado, declarativo y libre de side-effects.

De Especificación: En OCL no describimos cómo se implementan las cosas.

Tipificado: Cada expresión en OCL evalúa a un tipo, y se puede operar en función de los tipos definidos.

Declarativo: No hay expresiones imperativos, como la asignación.

Libre de side-effects: Lo expresado en OCL no puede modificar el sistema.

Modelos de Software

UML + OCL

Podemos utilizar OCL para

- Definición de invariantes

- Inicialización de propiedades de una clase

- Definición de reglas sobre cómo se calculan valores

- Definición de contratos

Modelos de Software

OCL - Invariantes

Los invariantes son restricciones de integridad sobre el sistema. Un invariante es, en esencia, una condición booleana que siempre debe ser verdadera.

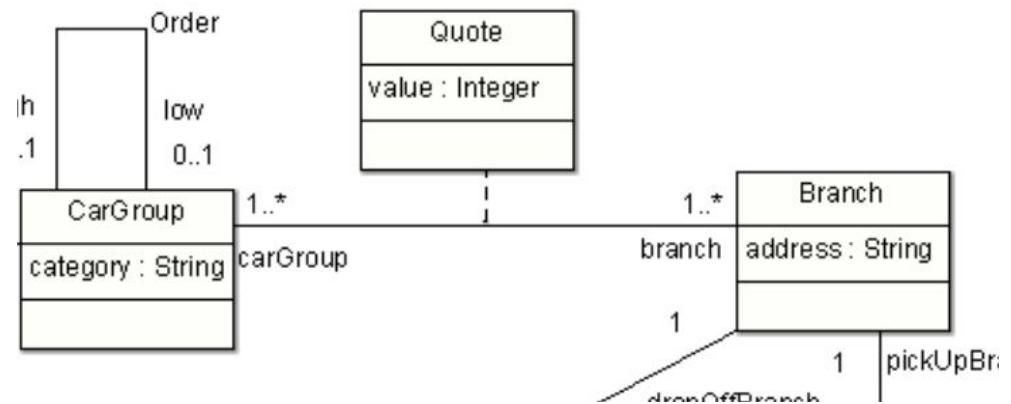
Modelos de Software

OCL - Invariantes

Los invariantes son restricciones de integridad sobre el sistema. Un invariante es, en esencia, una condición booleana que siempre debe ser verdadera.

Por ejemplo

`context Quote inv QuoteOverZerp: self.value > 0`



Modelos de Software

OCL - Invariantes

Los invariantes son restricciones de integridad sobre el sistema. Un invariante es, en esencia, una condición booleana que siempre debe ser verdadera.

Por ejemplo

```
context Quote inv QuoteOverZerp: self.value > 0
```

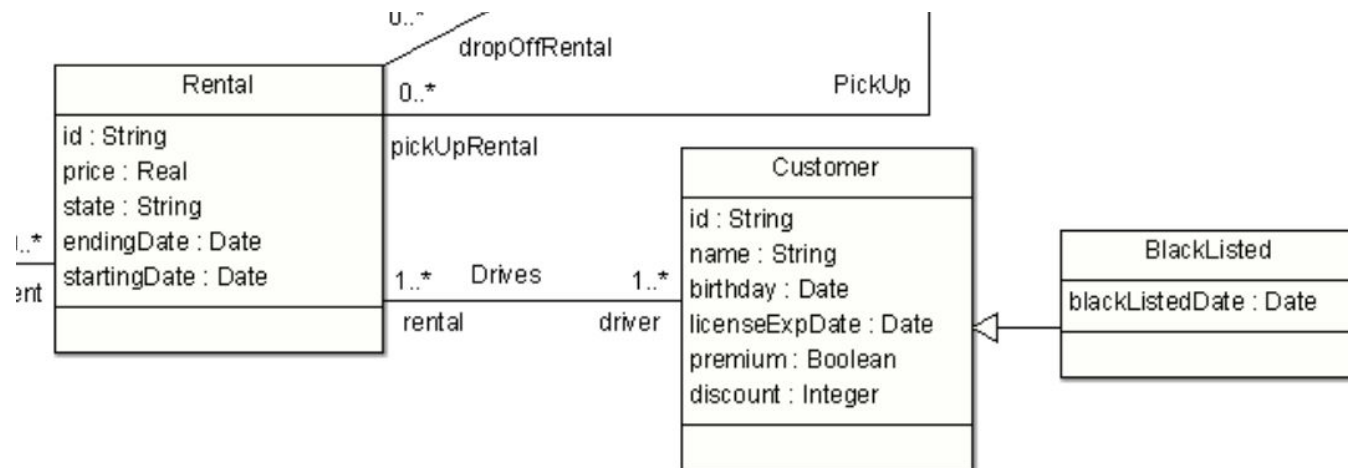
Todas las instancias de la clase Quote deben mantener esa condición booleana como verdadera.

Modelos de Software

OCL - Invariantes

Los invariantes pueden describir condiciones mucho más complejas

```
context BlackList inv NoRentalsBlackListed:  
self.rental->forAll( r | r.startDate < self.blackListedDate )
```

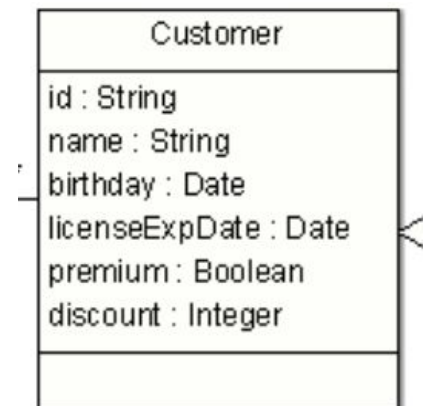


Modelos de Software

OCL - Inicialización

OCL se puede utilizar para definir cómo se deben inicializar los atributos de una clase cuando se crea una instancia de la misma.

`context Customer::premium: boolean init: false`



Modelos de Software

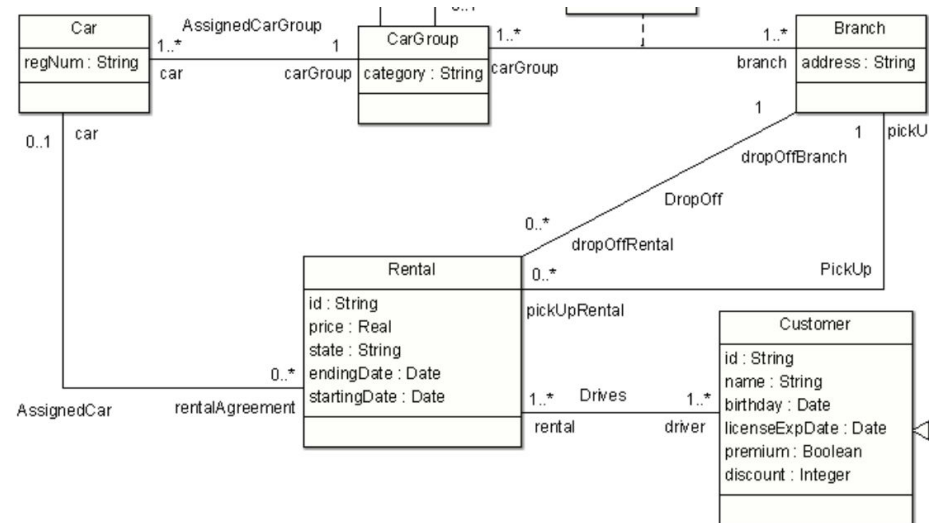
OCL - Elementos derivados

En OCL, un elemento derivado es aquel cuyo valor puede inferirse a partir del valor de otros elementos dentro del modelo.

context Customer::discount: integer

derive:

```
if not self.premium then
  if self.rental.car.carGroup->
    select(c | c.category='high')
    ->size()>=5
  then 15
  else 0 endif
else 30 endif
```



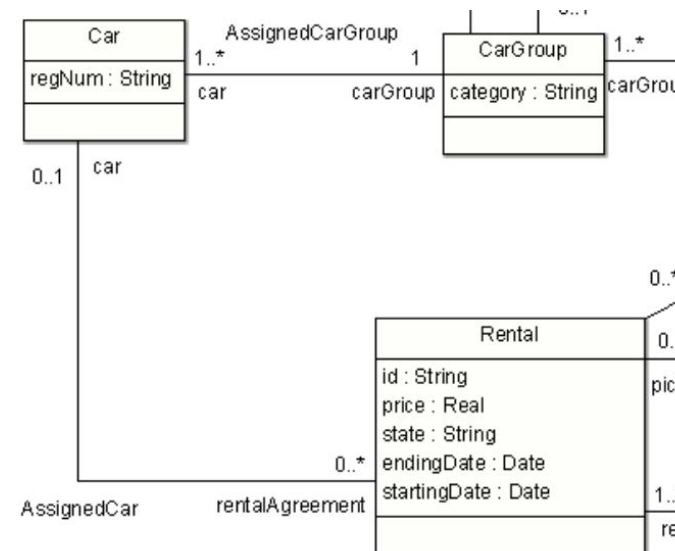
Modelos de Software

OCL - Consultas

En OCL podemos expresar consultas al modelo. Dichas consultas pueden ser utilizadas luego en otras expresiones de OCL.

context Car::mostPopular(): boolean

body: Car::allInstances()->forAll(c1 | c1<>self implies
c1.rentalAgreement->size() <=
self.rentalAgreement->size())



Modelos de Software

OCL - Contratos

Cada operación tiene asociada un contrato. Un contrato es un conjunto de condiciones booleanas dividido en dos, pre y post condiciones.

Las precondiciones definen las condiciones de verdad sobre los parámetros de entrada y el estado del sistema antes de ejecutar la operación.

Modelos de Software

OCL - Contratos

Cada operación tiene asociada un contrato. Un contrato es un conjunto de condiciones booleanas dividido en dos, pre y post condiciones.

Las postcondiciones definen las condiciones de verdad del sistema cuando la operación termina.

Modelos de Software

OCL - Contratos

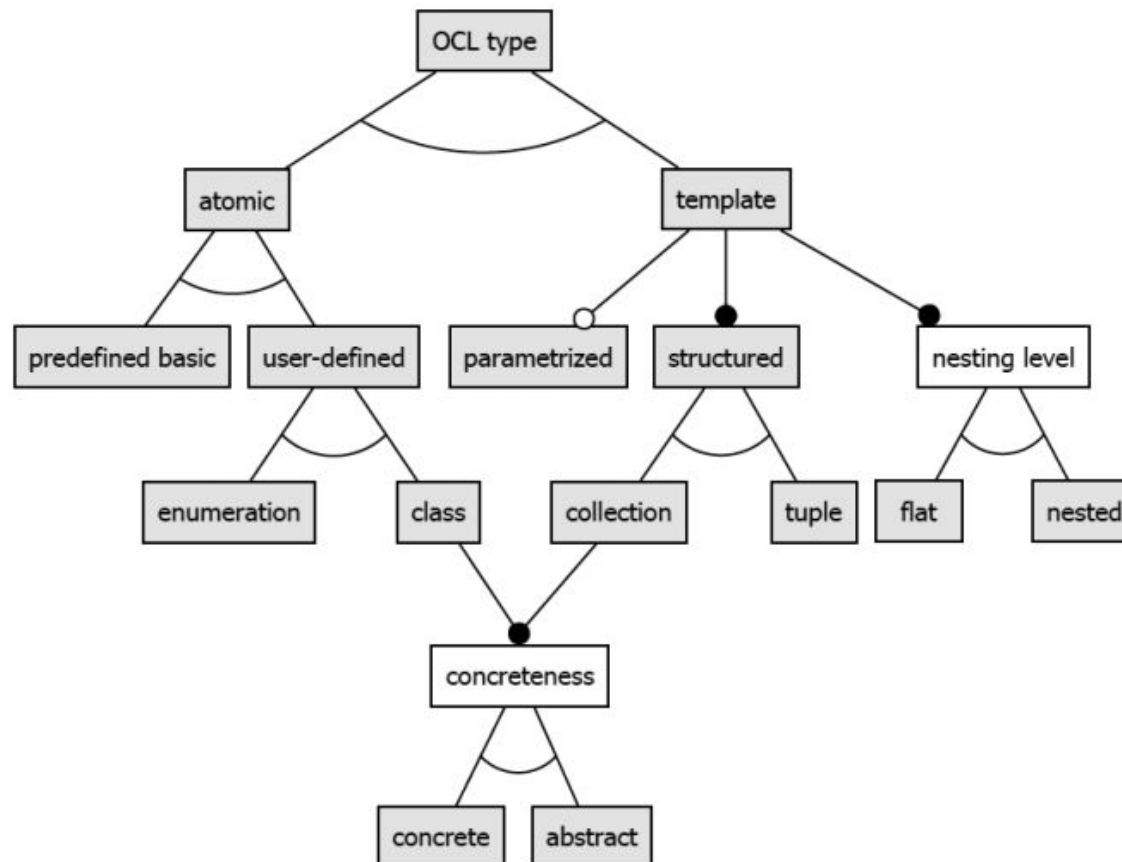
context Rental::newRental(id:Integer, price:Real, startingDate:Date, endingDate:Date, customer:Customer, carRegNum:String, pickupBranch: Branch, dropOffBranch: Branch)

pre: customer.licenseExpDate>endingDate

post: Rental.allInstances->one(r |
r.ocllsNew() and r.ocllsTypeOf(Rental) and
r.endingDate=endingDate and r.startingDate=startingDate and
r.driver=customer and r.pickupBranch=pickupBranch and
r.dropOffBranch=dropOffBranch and
r.car=Car.allInstances()->any(c | c.regNum=carRegNum))

Modelos de Software

OCL - Tipos



Modelos de Software

OCL - Eclipse

```
import ecore : 'http://www.eclipse.org/emf/2002/Ecore#/'

package ecore

context EModelElement
/**
 * Declare a helper operation to map an ok/warning verdict to ok/error.
 */
def: asError(verdict : Boolean) : Boolean = if verdict then true else null endif

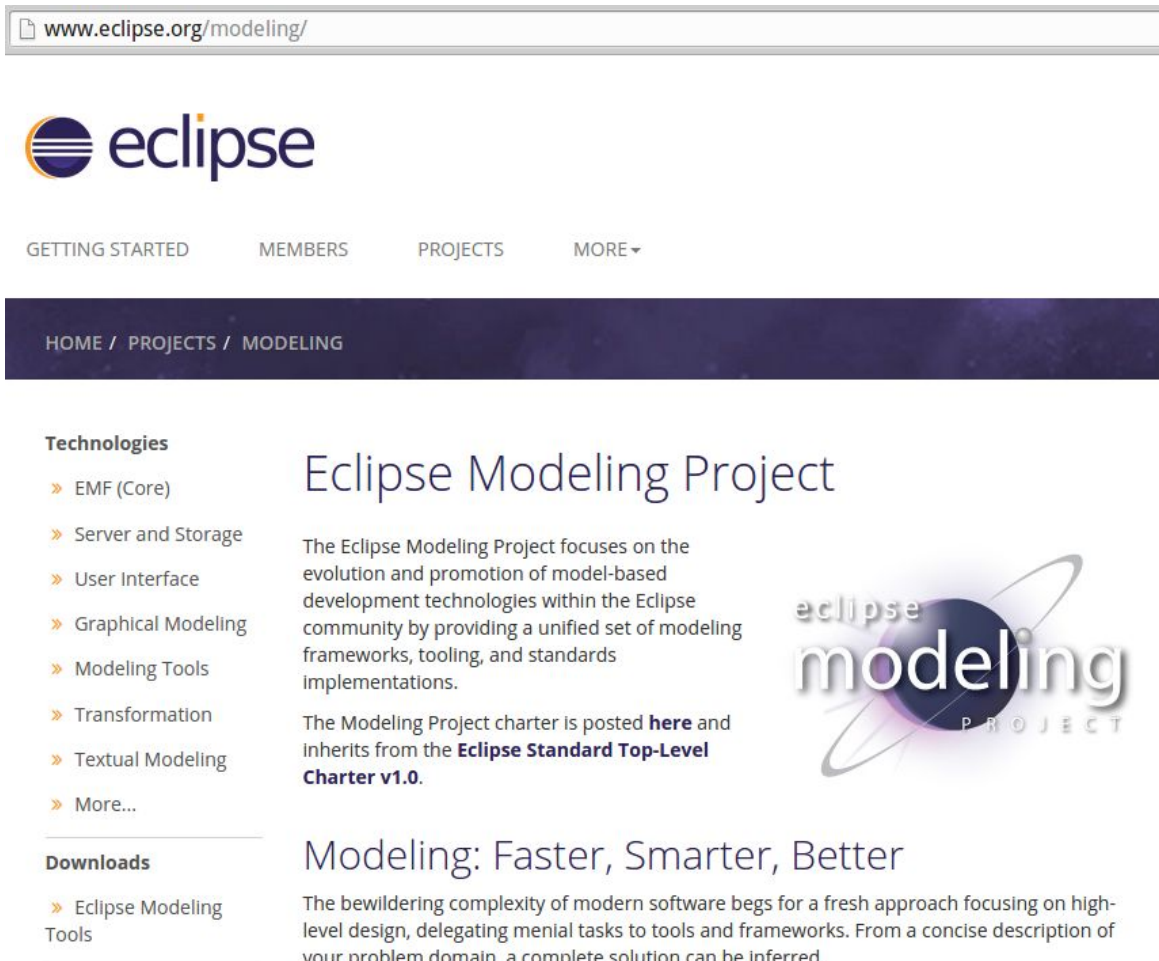
/**
 * Extra validation for EReference and EAttribute.
 */
context EStructuralFeature

/**
 * Declare a helper property to determine whether an EStructuralFeature has an OCL derivation.
 */
def: hasDerivation : Boolean =
  eAnnotations->select(source.startsWith('http://www.eclipse.org/emf/2002/Ecore/OCL'))->notEmpty()

/**
 * If a feature has an OCL derivation, it should be transient; otherwise it is not executed.
 */
inv DerivationIsTransient: hasDerivation implies transient
```



Modelos de Software

OCL - Eclipse



The screenshot shows the Eclipse Modeling Project website. At the top, there is a browser address bar with the URL www.eclipse.org/modeling/. Below the address bar is the Eclipse logo, which consists of a stylized globe icon followed by the word "eclipse" in a lowercase, sans-serif font. A navigation menu is located below the logo, with links for "GETTING STARTED", "MEMBERS", "PROJECTS", and "MORE" with a dropdown arrow. A dark blue banner below the navigation menu contains the breadcrumb "HOME / PROJECTS / MODELING". The main content area is divided into two columns. The left column has a "Technologies" section with a list of links: "EMF (Core)", "Server and Storage", "User Interface", "Graphical Modeling", "Modeling Tools", "Transformation", "Textual Modeling", and "More...". Below this is a "Downloads" section with a link for "Eclipse Modeling Tools". The right column features the "Eclipse Modeling Project" title, a paragraph describing the project's focus on model-based development technologies, and a link to the "Eclipse Standard Top-Level Charter v1.0". Below this is another section titled "Modeling: Faster, Smarter, Better" with a paragraph explaining the project's goal of simplifying software development through high-level design and tool delegation. To the right of the text is a logo for the Eclipse Modeling Project, featuring the words "eclipse modeling" in a stylized font with a glowing sphere and orbital lines behind them, and the word "PROJECT" in a smaller font below.

www.eclipse.org/modeling/

 eclipse

GETTING STARTED MEMBERS PROJECTS MORE ▾

HOME / PROJECTS / MODELING

Technologies

- » [EMF \(Core\)](#)
- » [Server and Storage](#)
- » [User Interface](#)
- » [Graphical Modeling](#)
- » [Modeling Tools](#)
- » [Transformation](#)
- » [Textual Modeling](#)
- » [More...](#)

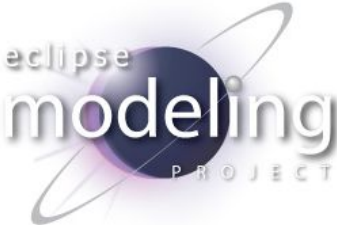
Downloads

- » [Eclipse Modeling Tools](#)

Eclipse Modeling Project

The Eclipse Modeling Project focuses on the evolution and promotion of model-based development technologies within the Eclipse community by providing a unified set of modeling frameworks, tooling, and standards implementations.

The Modeling Project charter is posted [here](#) and inherits from the **Eclipse Standard Top-Level Charter v1.0**.



Modeling: Faster, Smarter, Better

The bewildering complexity of modern software begs for a fresh approach focusing on high-level design, delegating menial tasks to tools and frameworks. From a concise description of your problem domain, a complete solution can be inferred.

Modelos de Software

Java Modeling Language (JML)

Modelos de Software

Java Modeling Language (JML)

Al igual que OCL, JML permite declarar condiciones booleanas que el sistema debe respetar.

Modelos de Software

Java Modeling Language (JML)

```
//@ requires x >= 0.0;
/*@ ensures JMLDouble
    @         .approximatelyEqualTo
    @         (x, \result * \result, eps);
    @*/
public static double sqrt(double x) {
    /*...*/
}
```

Modelos de Software

Java Modeling Language (JML)

```
//@ requires x >= 0.0;
/*@ ensures JMLDouble
    @         .approximatelyEqualTo
    @         (x, \result * \result, eps);
    @*/
public static double sqrt(double x) {
    /*...*/
}
```

```
//@ requires (* x is positive *);
/*@ ensures (* \result is an
    @         approximation to
    @         the square root of x *)
    @         && \result >= 0;
    @*/
public static double sqrt(double x) {
    return Math.sqrt(x);
}
```

Modelos de Software

Java Modeling Language (JML)

```
public class Person {
    private String name;
    private int weight;

    /*@ also
       @ ensures \result != null
       @   && (* \result is a displayable
       @     form of this person *);
       @*/
    public String toString() {
        return "Person(\"" + name + "\", "
            + weight + ")";
    }
}
```

Modelos de Software

Java Modeling Language (JML)

```
/*@ requires a != null
   @         && (\forall int i;
   @         0 < i && i < a.length;
   @         a[i-1] <= a[i]);
   @*/
int binarySearch(int[] a, int x) {
    // ...
}
```

Modelos de Software

Java Modeling Language (JML)

```
/*@ refine "Person.java";
```

```
public class Person {  
    private /*@ spec_public non_null @*/  
        String name;  
    private /*@ spec_public @*/  
        int weight;  
  
    /*@ public invariant !name.equals("")  
        @          && weight >= 0; @*/
```

```
    /*@ also  
    /*@ ensures \result != null;  
    public String toString();
```

```
    /*@ also  
    /*@ ensures \result == weight;  
    public /*@ pure @*/ int getWeight();
```

```
    /*@ also  
        @ requires kgs >= 0;  
        @ requires weight + kgs >= 0;  
        @ ensures weight == \old(weight + kgs);  
    @*/
```

```
    public void addKgs(int kgs);
```

```
    /*@ also  
        @ requires n != null && !n.equals("");  
        @ ensures n.equals(name)  
        @   && weight == 0; @*/  
    public Person(String n);
```

```
}
```


Modelos de Software

OpenJML

Downloads

Documentation

Try Online

Wiki

Features

Papers

About

Does your program do what it is supposed to do?

OpenJML is a program verification tool for Java programs that allows you to check the specifications of programs annotated in the Java Modeling Language.

[Download OpenJML](#)

```
// Can you spot the two errors  
// in this program?
```

```
public class MaybeAdd {  
    //@ requires a > 0;  
    //@ requires b > 0;  
    //@ ensures \result == a+b;  
    public static int add(int a, int b){  
        return a-b;  
    }  
  
    public static void main(String args[]){  
        System.out.println(add(2,3));  
    }  
}
```



Modelos de Software



OpenJML (JML)

Eclipse plug-in for OpenJML

This page provides installation instructions and a brief overview of the functionality of the OpenJML Eclipse plugin. More detailed information about the plugin is available in the Help documentation that is installed with the plugin.

Installing

OpenJML provides a standard Eclipse plug-in update site at <http://jmlspecs.sourceforge.net/openjml-updatesite>. The instructions for installation are given [here](#).

[TBD - Add a description of installation using the dropins folder]

Features

[TBD - this section needs review and updating]

MENUBAR AND TOOLBAR ITEMS

The Menubar and toolbar contain these additional items:

- The JML menubar entry with its submenu items.
- A toolbar item marked by the JML logo (a yellow coffee cup), which initiates type-checking of a Java or JML file.
- A toolbar item marked ESC, which initiates static checking of selected files, classes or methods.
- A toolbar item marked PAC, which initiates compilation for runtime assertion checking.