

# The rCOS UML profile

Volker Stolz

`vs@iist.unu.edu`



United Nations  
University

**UNU-IIST**

International Institute for  
Software Technology



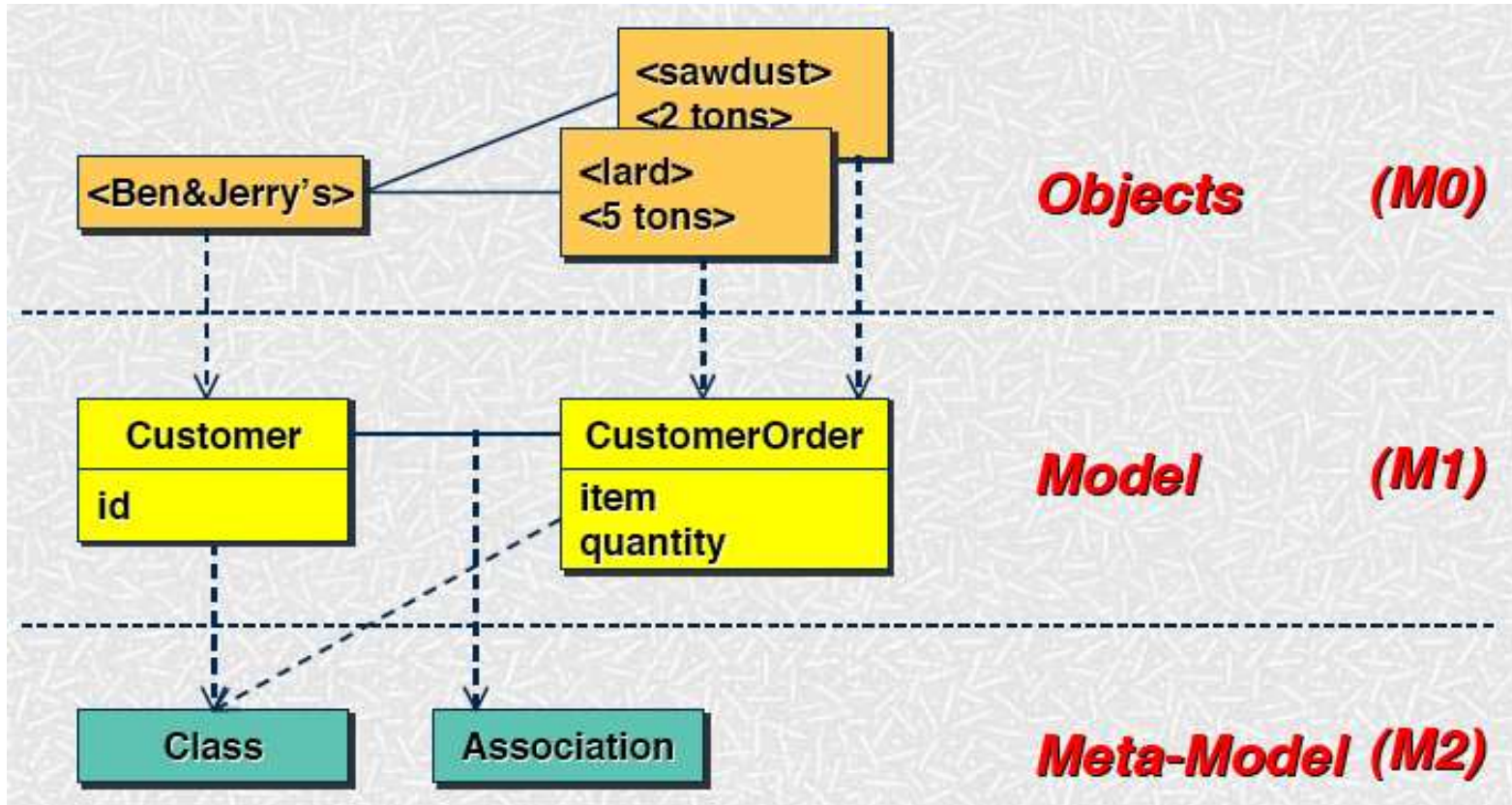
# Overview

- Introduction to UML profiles
  - Stereotypes / tagged values
- The rCOS UML profile
- The rCOS Modeling Tool
- Summary

# Introduction to UML profiles

- What is UML?
  - not only diagrams, also *models!*
- Most familiar: class diagram (model)
- Meta-modeling: how would you model UML itself?
  
- Based on:
  - “Object Modeling with UML: Advanced Modeling” by OMG
  - <http://www.omg.org>

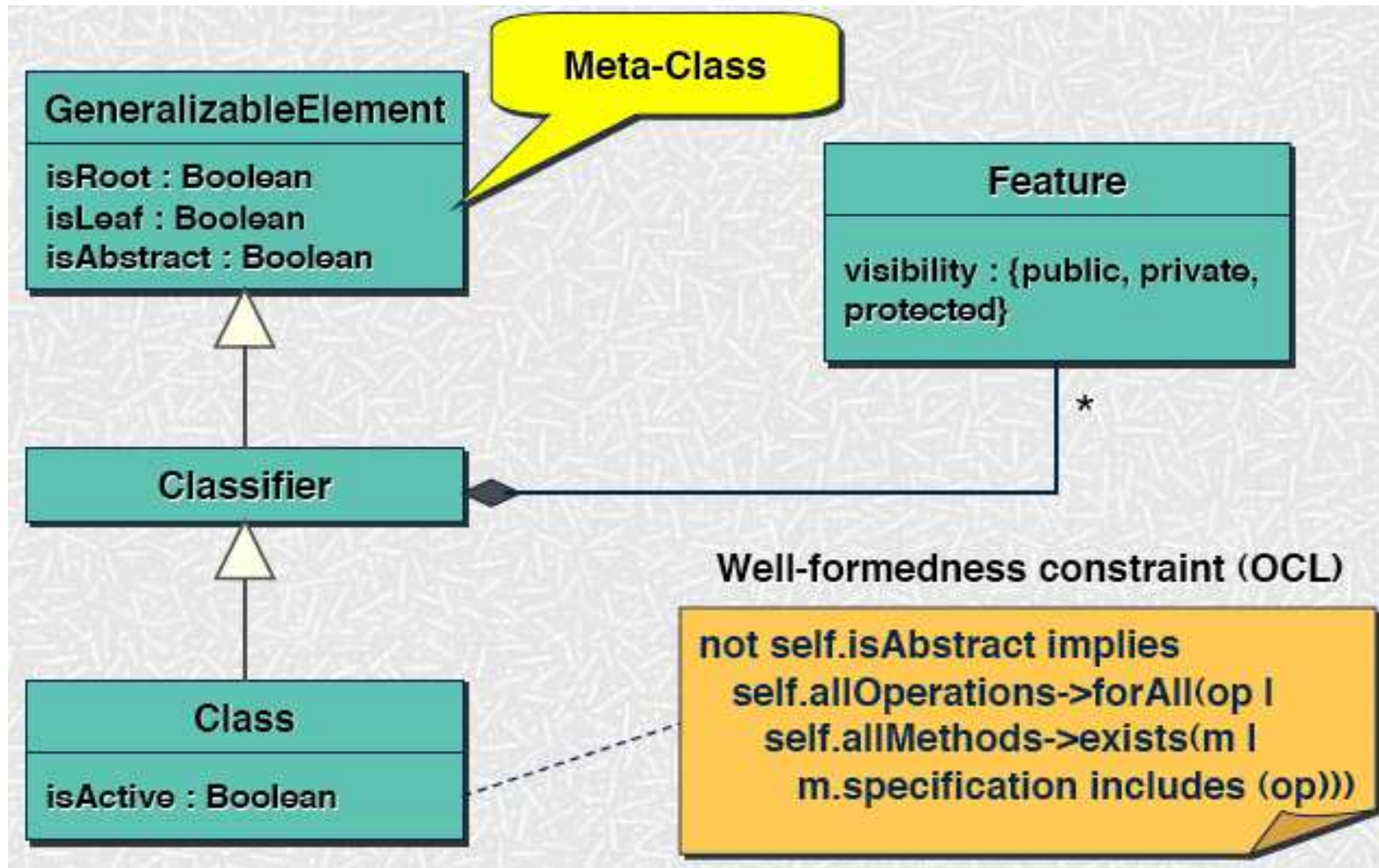
# Meta-Models



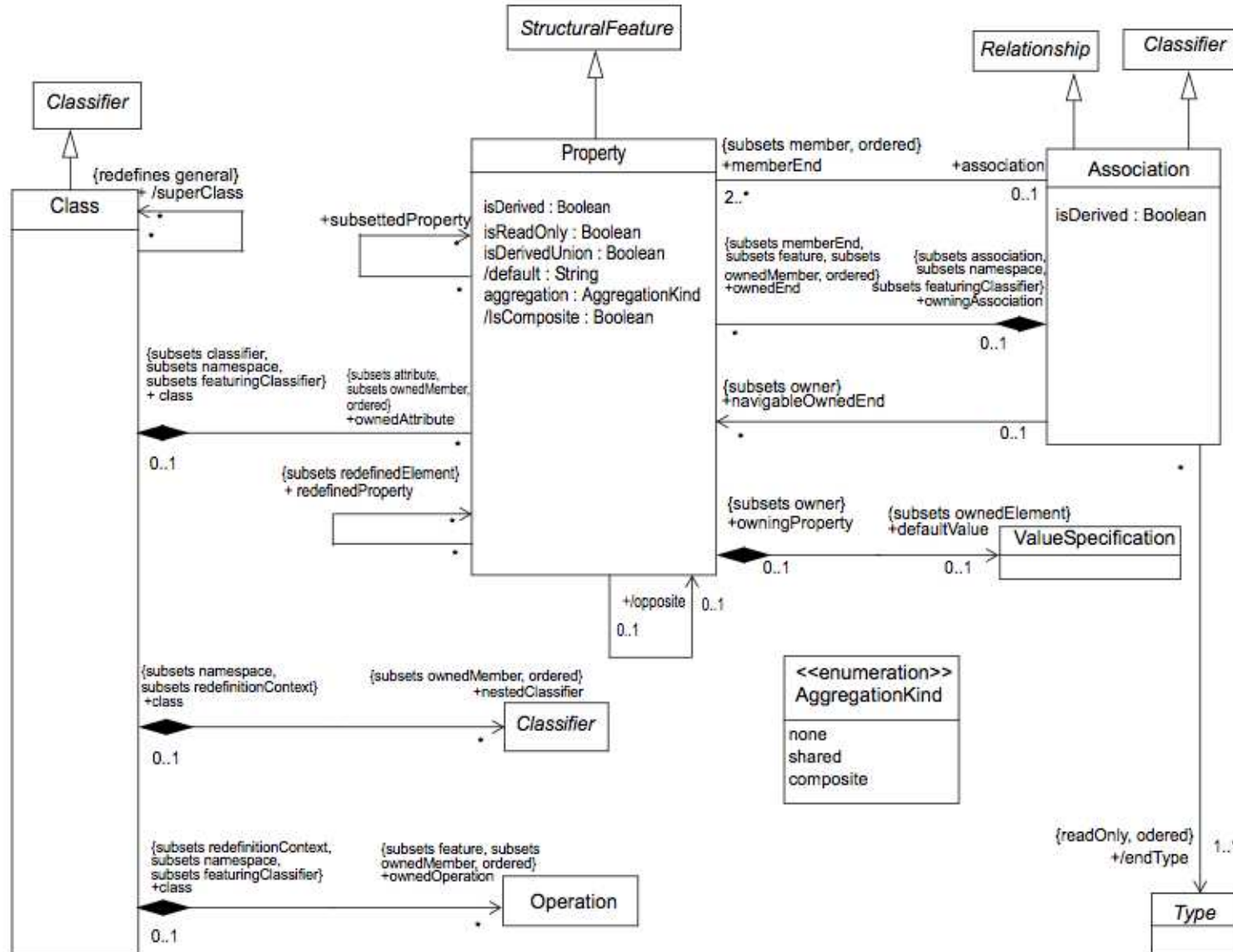
- framework developers *edit* the meta-model
- application developers *use* the meta-model

# Meta-Models

Expressed using a very small subset of UML:



# UML modelled in UML



# UML Extension Mechanism

- Core UML is limited
- without extension mechanisms, the current standard would be even more complex

## UML compared with programming language

- PL defines small set of core concepts and libraries
- programmer extends vocabulary through new names for methods, classes, etc
- UML:
  - vocabulary extension through introduction of new modeling elements (specific classes)
  - stereotypes apply on meta-level: do not represent phenomena of the system, but of the UML model

# The Three Basic Mechanisms

## Stereotypes

- used to refine meta-classes or other stereotypes
- similar to subclassing

## Tagged Values

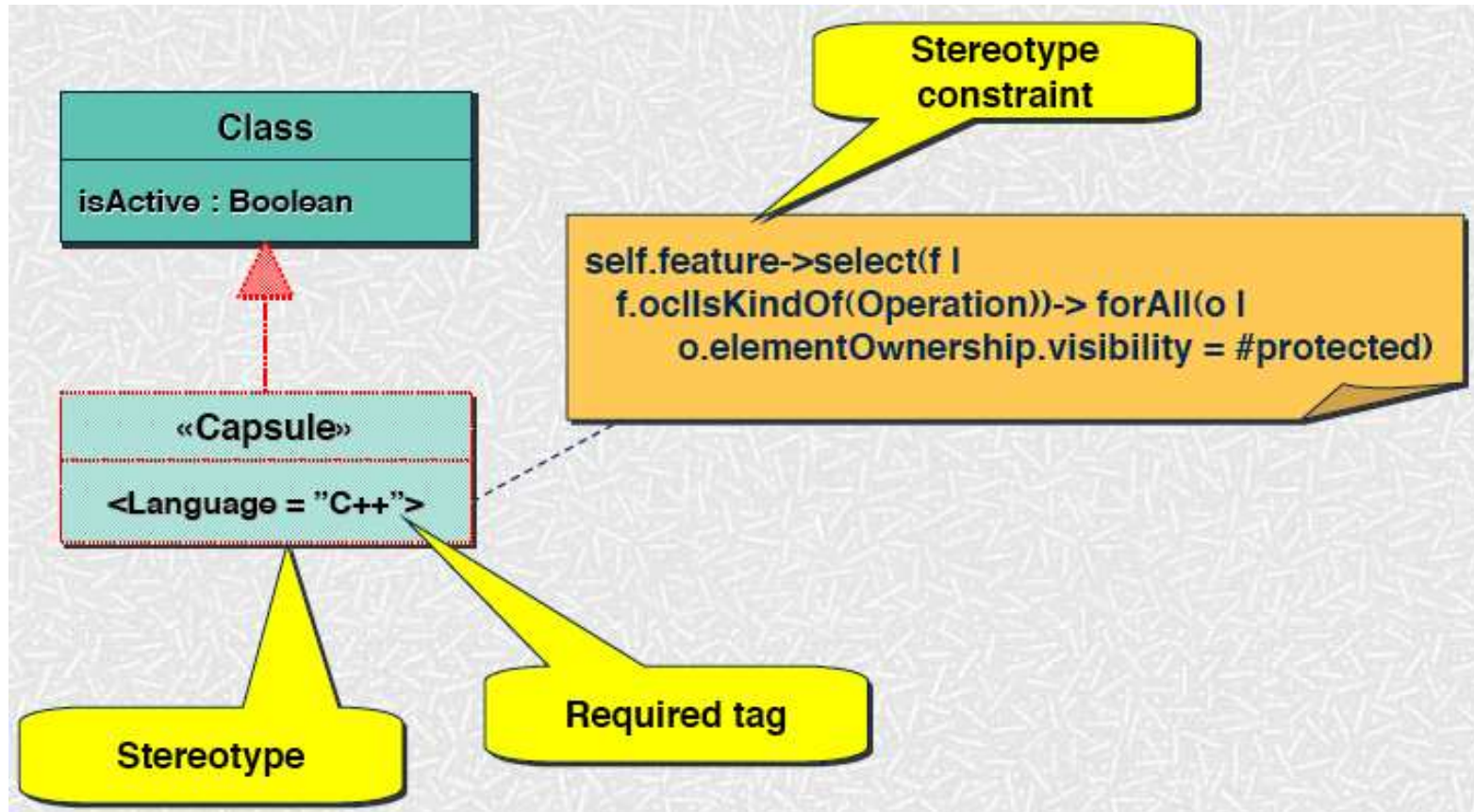
- introduces new attribute with given type
- can be attached to any meta-class or stereotype

## Constraints

- predicates (e.g., OCL expressions) that reduce semantic variation
- can be attached to any meta-class or stereotype



# Example: Stereotyping a Class



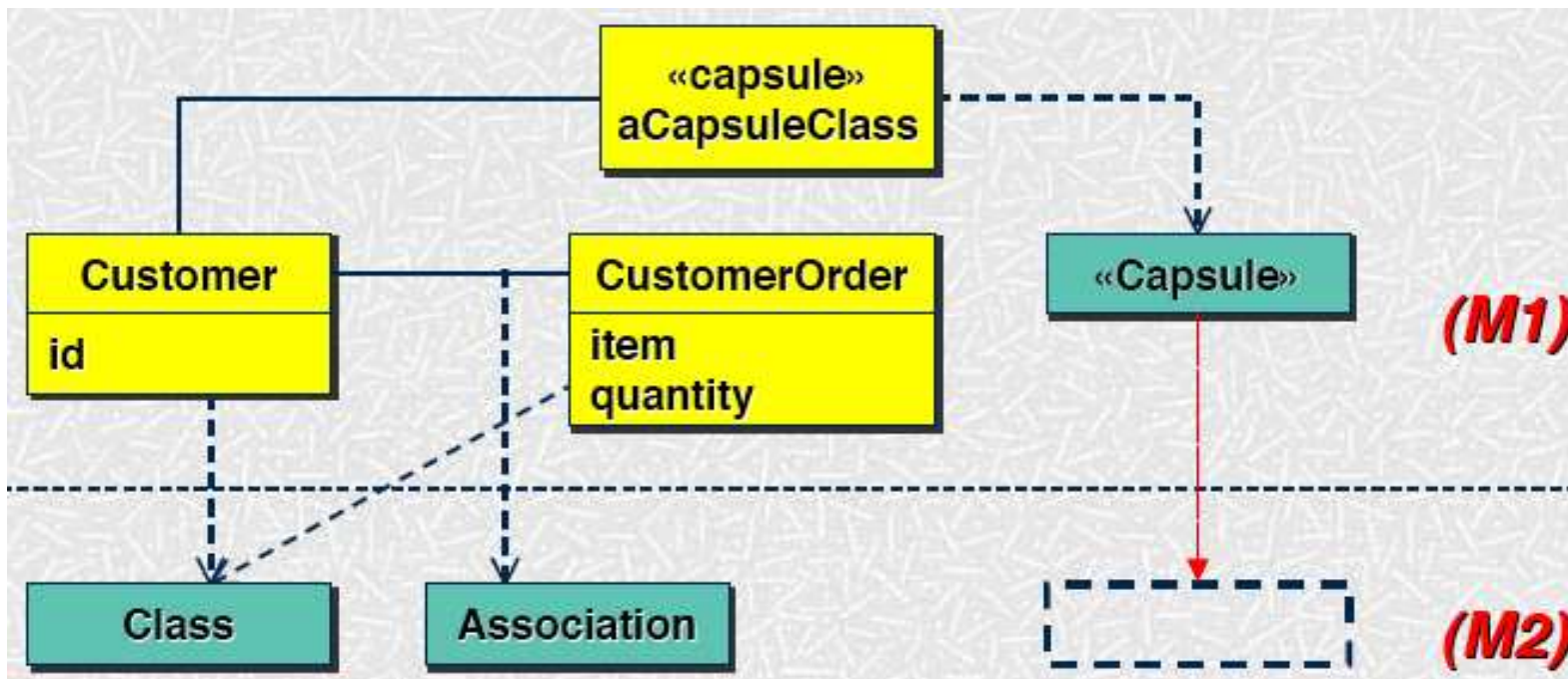
For *class diagrams*, “stereotyping” overlaps with OO-modeling.

**Important:** just typed markup, meaning depends on consumer

# Extensibility Method

Refinements are specified at the Model (M1) level but apply to the Meta-Model level (M2)

- does not require “meta-modeling” CASE tools
- can be exchanged with models



# Stereotypes

- Used to define derivative modeling concepts based on existing generic modeling concepts
- Defined by:
  - base (meta-)class = UML meta-class or stereotype
  - constraints
  - required tags (0..\*)
  - icon

# Tagged Values

## Tagged values...

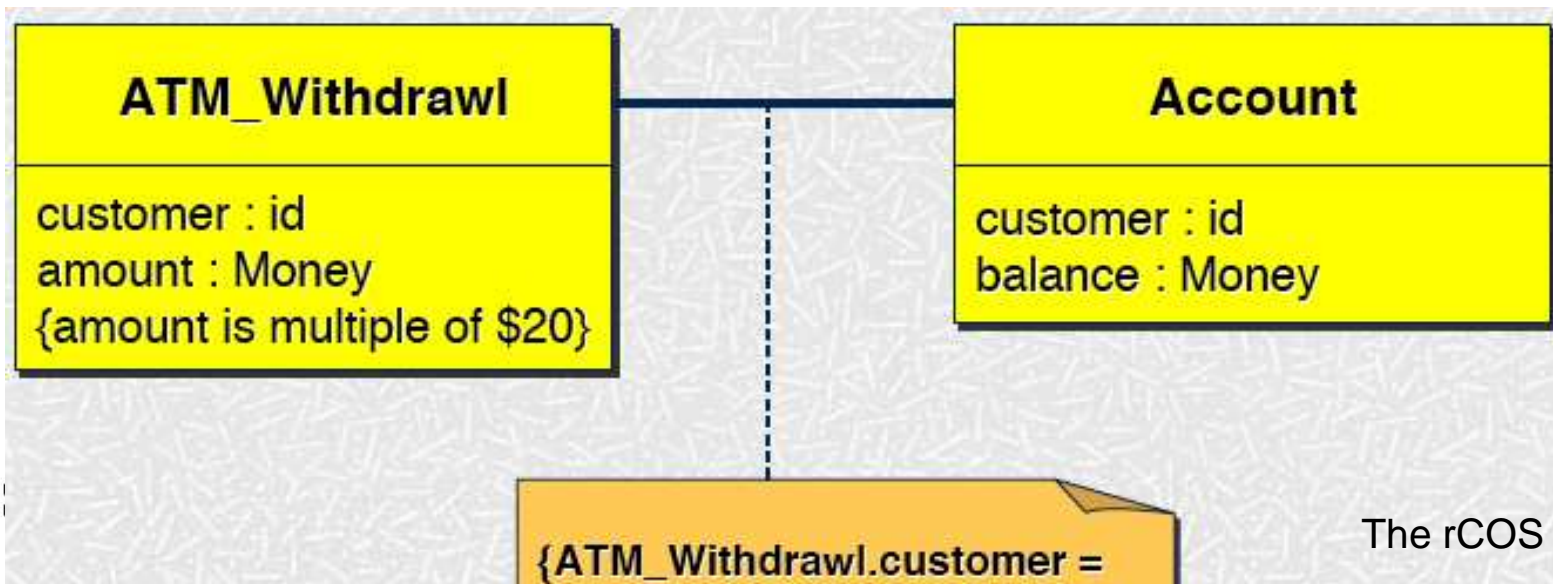
- consist of a tag and value pair
- used to model stereotype attributes
- have arbitrary domain-specific semantics

## Examples

- name of the creator of a class
- version number, date of last change
- icon for graphical representation

# Constraints

- formal or informal expressions
- must not contradict inherited base semantics
- specified
- Constraint Notation
  - Enclosed in braces “{...}”
  - Can appear in various places in a model
- Problem: must be enforced by tool



# UML Profile

A package of related specializations of general UML concepts that capture domain-specific variations and usage patterns

⇒ A domain-specific interpretation of UML

## Existing profiles by OMG:

- EDOC
- Real-Time
- CORBA
- UMLSec

# UML Profile

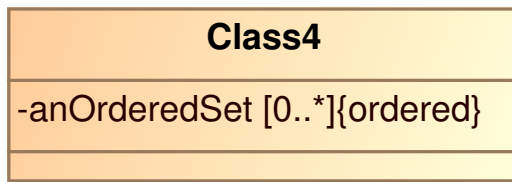
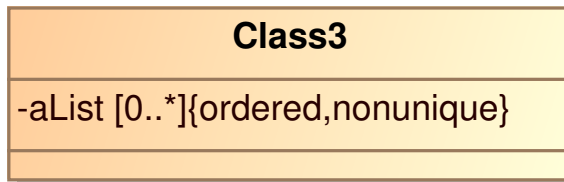
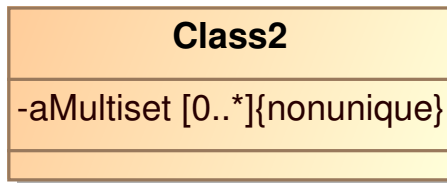
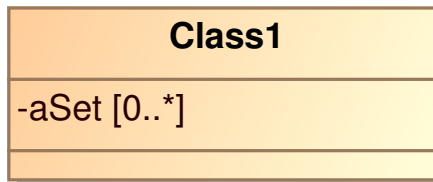
- defines permissible stereotypes and their attributes/constraints
- references a meta-model
- graphical representation: profile diagram  
similar to class diagram:
  - entities to apply stereotypes on are classes from the meta-model  
(meta-classes)

## Class Diagrams

- “useful” features for programming already present: attributes, subclasses, interfaces
- so: meta-information to be consumed by *others*:
  - informal semantic markup for humans
  - model-processing/transformation

**Example: array/set/list in class diagram**

# Examples



The screenshot shows the Properties window for the attribute `-aSet [0..*] [Class1]`. The window title is `-aSet [0..*] [Class1]`. The main area is titled `aSet` and contains a table of properties. The `Is Unique` property is checked.

Property	
Name	aSet
Qualified Name	Class1::aSet
Type	
Type Modifier	
Visibility	private
Default Value	
Applied Stereotype	
Multiplicity	0..*
Is Read Only	<input type="checkbox"/> false
Is Static	<input type="checkbox"/> false
Aggregation	none
Is Derived	<input type="checkbox"/> false
Is Derived Union	<input type="checkbox"/> false
Is Ordered	<input type="checkbox"/> false
Is Unique	<input checked="" type="checkbox"/> true
Subsetted Property	
Redefined Property	
To Do	

**Name**  
The name of the NamedElement.

# Stereotypes on other modeling elements

- use case: different types of actors
- state machine: timing constraints
- generally:
  - construct typed tuples of UML elements
  - package + «profile» = profile

## The rCOS UML profile

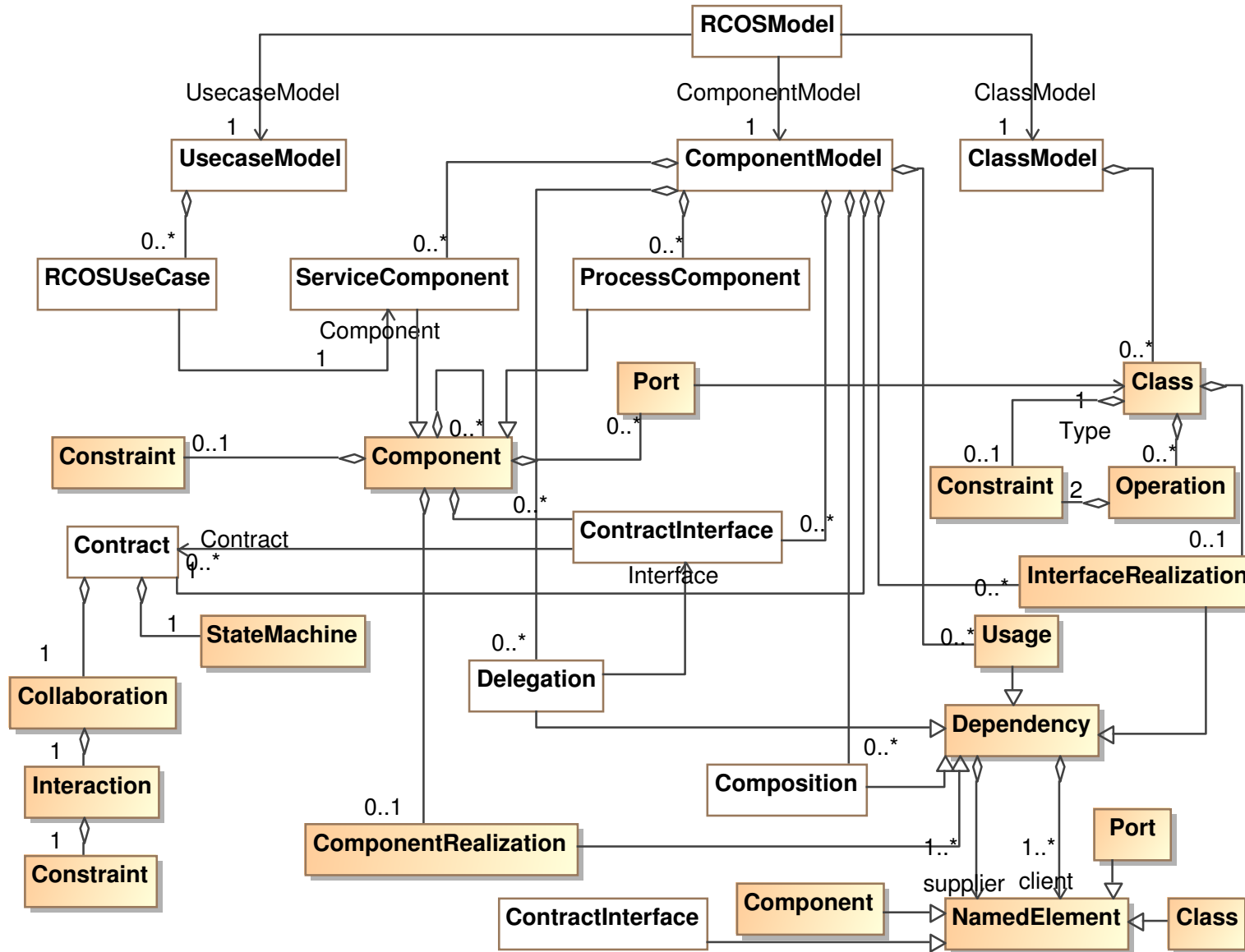
## The rCOS Modeler

- Design decision: use UML
- Differences between rCOS and OO/UML
- rCOS profile package

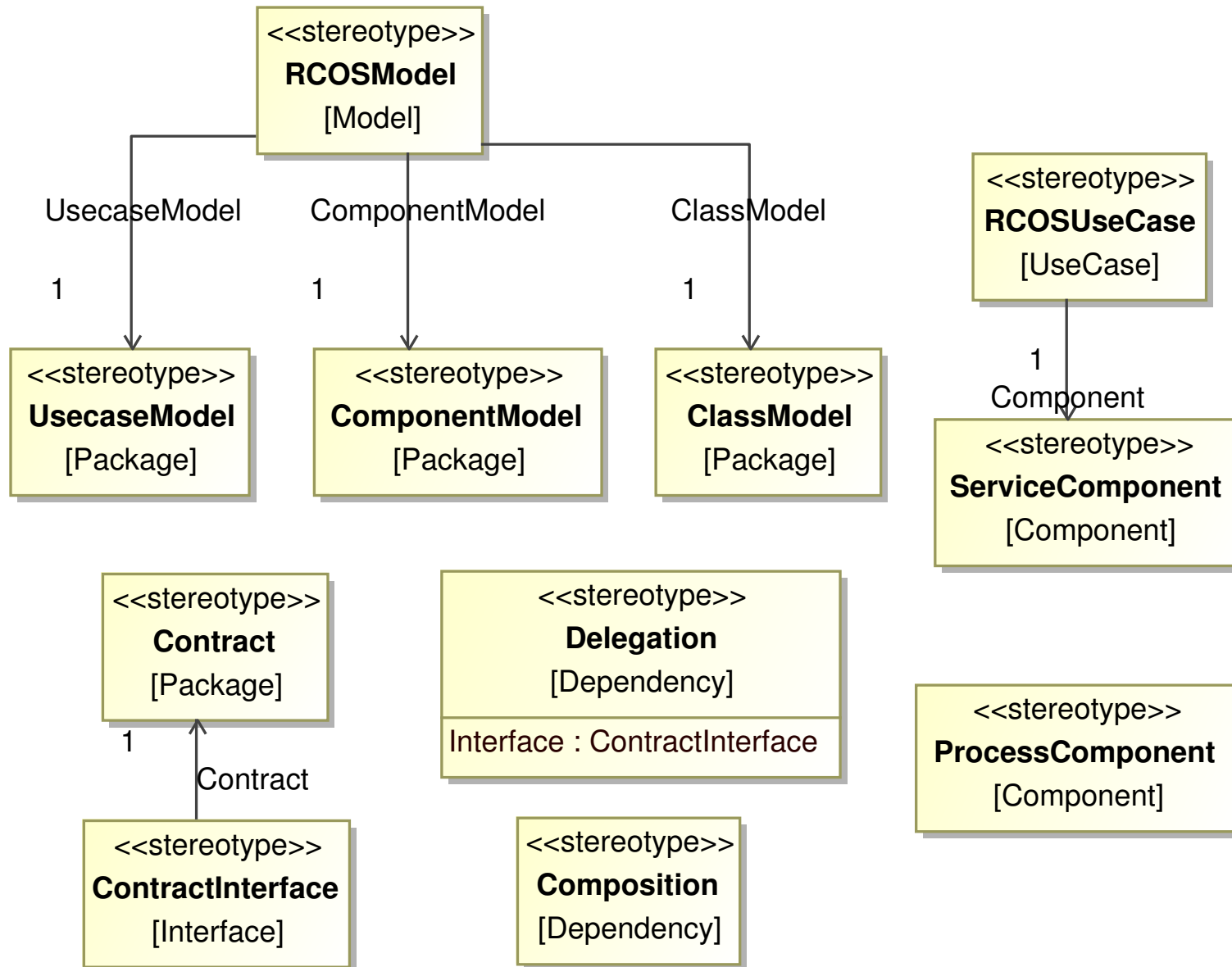
# Design Decision

- Advantages of using UML
  - many artifacts overlap: (classes/methods/associations, state machines, sequence diagrams)
  - existing tool support for UML modeling
- Disadvantages of using UML
  - allows incomplete specifications
  - not every UML model an rCOS model
  - subtle differences may confuse the casual user
  - some datastructures convoluted (e.g. in state machines)

# rCOS Data Model



# ... as a Profile Diagram:



## Stereotypes...

- allow extension of UML through *specialization*
- *tagged values* introduce attributes (very much like OO modeling)
- may use *constraints* in OCL
- have user-defined semantics
- may be processed by other tools

## Profiles...

- define set of stereotypes and tagged values
- existing, standardized profiles for many purposes
- graphically specified similar to class diagrams
- can be complex, and require documentation