

# CHAPTER 4: OO MODELING WITH UML

## Software Engineering

### DOCENCIA VIRTUAL

**Finalidad:**

Prestación del servicio Público de educación superior (art. 1 LOU)

**Responsable:**

Universitat Politècnica de València.

**Derechos de acceso, rectificación, supresión, portabilidad, limitación u oposición al tratamiento conforme a políticas de privacidad:**

<http://www.upv.es/contenidos/DPD/>

**Propiedad intelectual:**

Uso exclusivo en el entorno de aula virtual.

Queda prohibida la difusión, distribución o divulgación de la grabación de las clases y particularmente su compartición en redes sociales o servicios dedicados a compartir apuntes.

La infracción de esta prohibición puede generar responsabilidad disciplinaria, administrativa o civil







UNIVERSITAT  
POLITÀCNICA  
DE VALÈNCIA



# Contents

1. Motivation.
2. OO Modelling.
  1. View of an OO System.
3. The UML Notation.
  1. Class Diagrams (Part 1)
  2. Use Cases Diagrams (Part 2)

# Bibliography

-  Booch, G., Rumbaugh, J., Jacobson, I., UML. The Unified Modelling Language. UML 2.0 2ª Edición. Addison-Wesley, 2006
-  Stevens, P., Pooley, R. Utilización de UML en Ingeniería del Software con Objetos y Componentes. 2ª Edición. Addison-Wesley Iberoamericana 2007 Ingeniería del Software. (8ª ed.). Addison-Wesley, 2008
-  Fowler, M., UML Distilled. Addison-Wesley Object Technology Series, 2003
-  <https://www.uml.org/>

# Motivation

## What is a model?

“A model is a simplified view of reality”

## Why do we model?

To better understand the system under we are developing

- Models help **visualizing** how a system is or we want it to be.
- Models **specify** the structure and behavior of a system
- Models provide templates to guide the **construction** of a system
- Models **document** the decisions that we have adopted

# Motivation (OO Modelling)

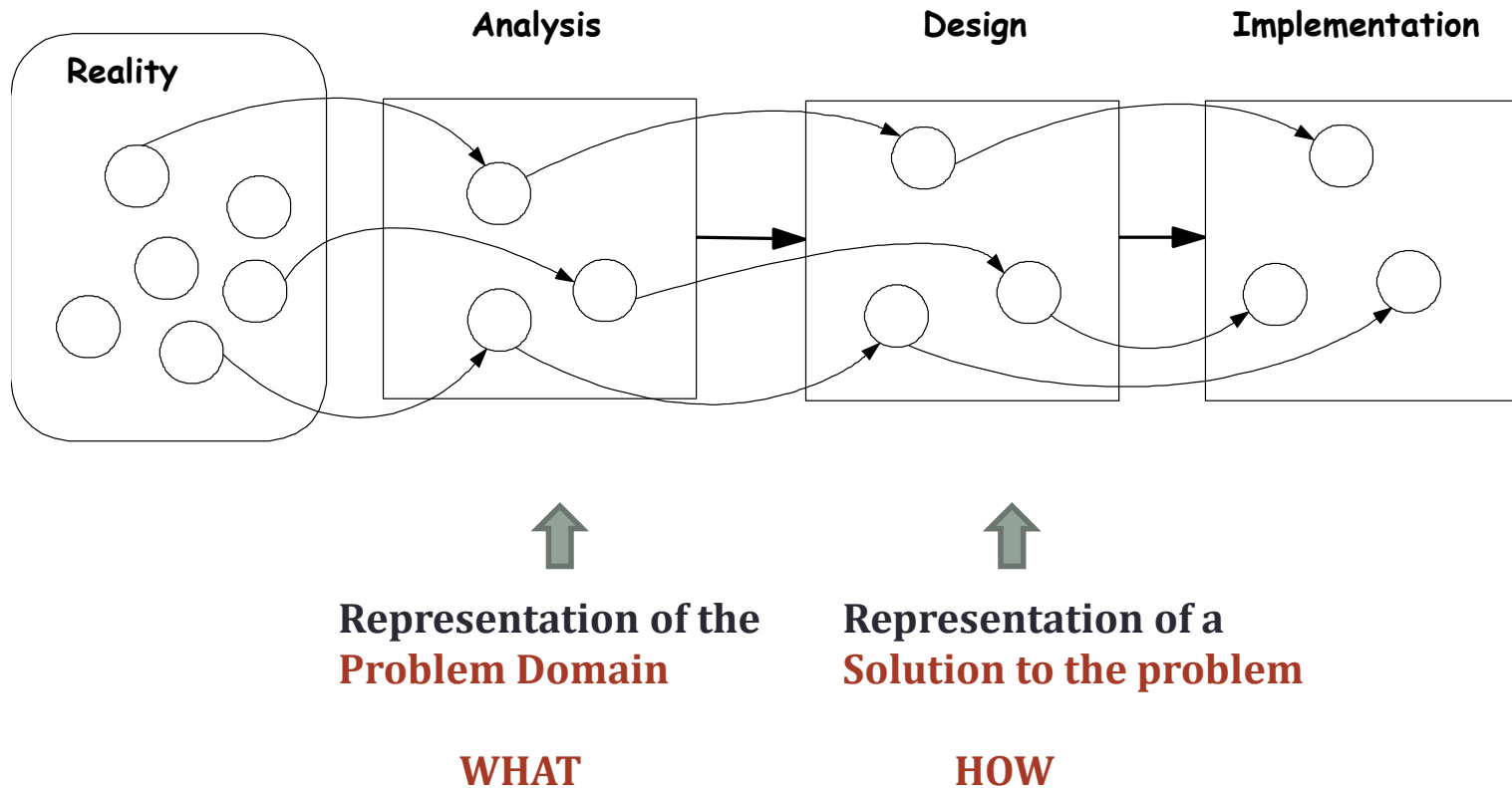
- OO Programming languages appear.
- The use of these languages requires a new viewpoint with respect to analysis and design.
- First OO analysis and design methods appear.

# Motivation (OO Modelling)

- OO methods represent requirements in terms of objects and the services they offer.
- OO methods are more “natural” than traditional ones:
  - Functions/processes vs objects.

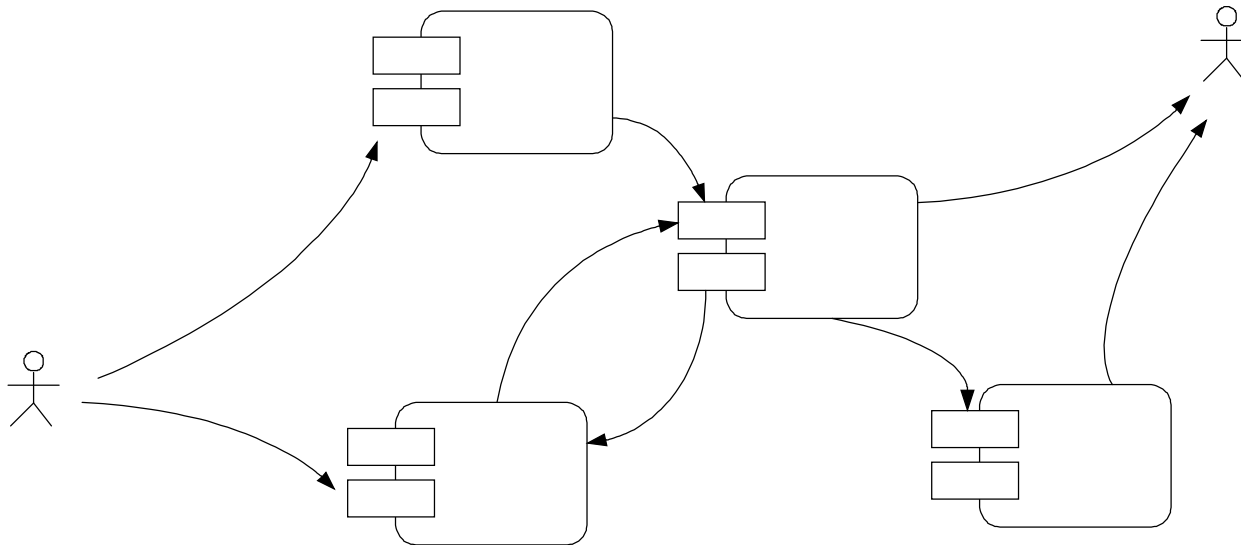
## OO Methods: Continuity between models

- In OO:



# 2 View of a Software System

**Static View + Dynamic View**





# Static View

- Object:
  - Entity that exists in the real world.
  - Have identity and are differentiated.
  - Examples:
    - The bill 2003/0010
    - The plane with plate number 123
    - A customer
    - The plane with plate number 345

# Static view

- Object Classes: Describe a collection of objects with:
  - Same properties.
  - Shared Behavior.
    - The plane with plate number 123
    - The plane with plate number 345

Abstraction

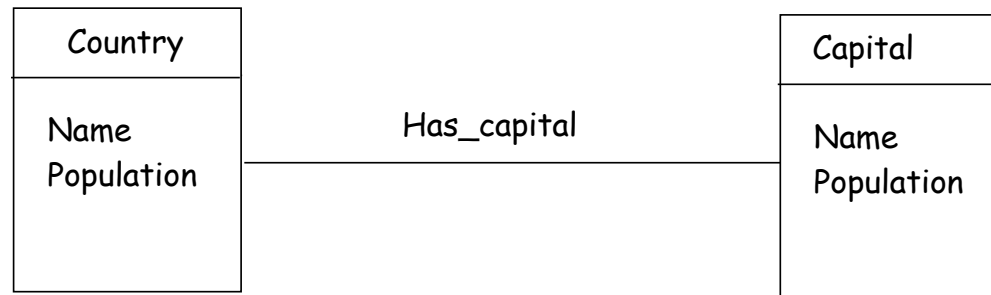


Eliminate differences among objects  
to keep shared aspects.

Plane

# Static View: Associations

- Association: Allows linking or connecting objects of different classes.
- Example: A country has only one capital.



# Static View

- Static Aspect: Describes the static structure of a system and its interrelationships.

	Intra-objects	Inter-objects
Static Aspect	Object classes. Attributes Operations	Association Generalization ....

# Dynamic View

- Objects communicate by means of invocation of operations on other objects.
- The dynamic view describes the aspects of a system that change over time:
  - Interactions between objects.
  - Possible states of an object.
  - Transitions between states.
  - What events are produced.
  - What operations are executed.

# Static/Dynamic Views

- Static View: Structure and interrelationships.
- Dynamic View: Aspects that change overtime.

	Intra-object	Inter-objects
Static Aspect	Object classes. Attributes Operations	Association Generalization ....
Dynamic Aspect	State Transition Diagrams	MSCs ....

## 4 The UML Language

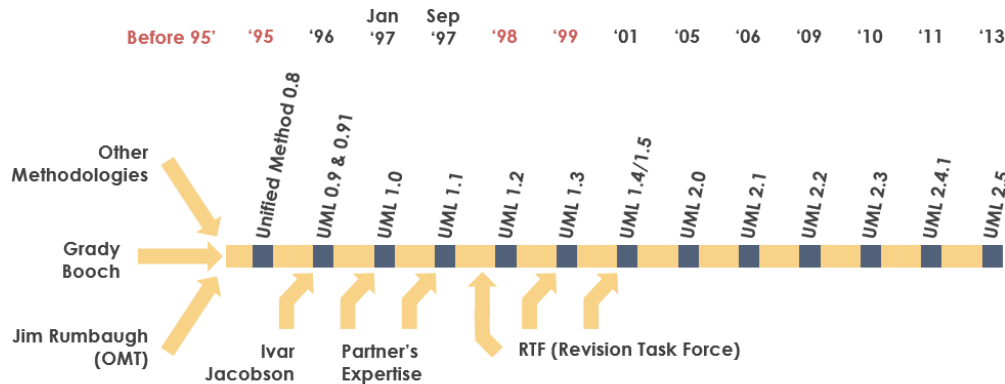
- UML = Unified Modeling Language
- UML: A general purpose language for OO modelling
- Starting Point:
  - Many OO methods with different notations.
  - Learning and tool construction inconvenients.
  - A Uniform notation needed.

# UML History

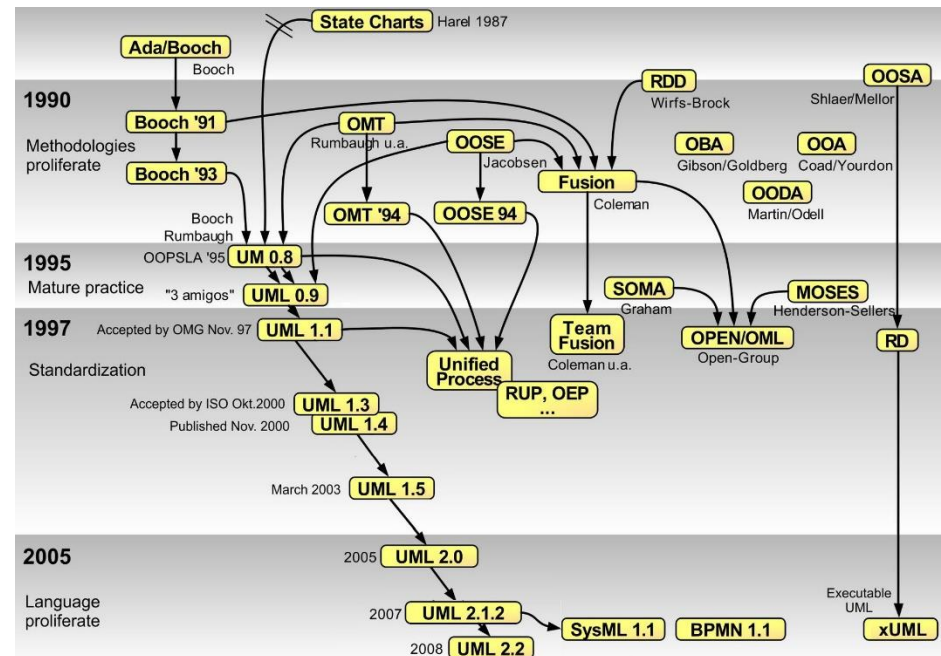
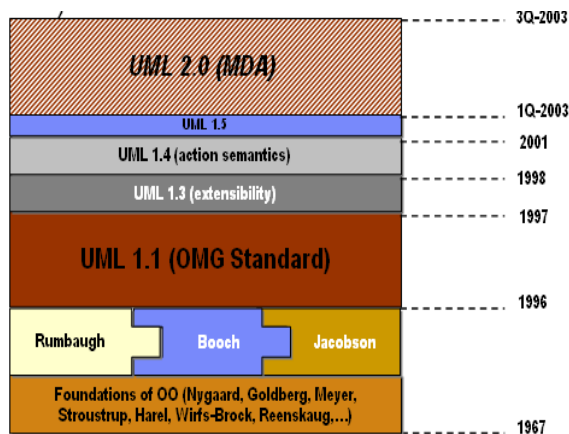
- Started as the “unified method” with the participation of J. Rumbaugh and G. Booch in 1995. The same year I. Jacobson is incorporated.
- Partners in Rational Software, CASE tool Rational Rose.



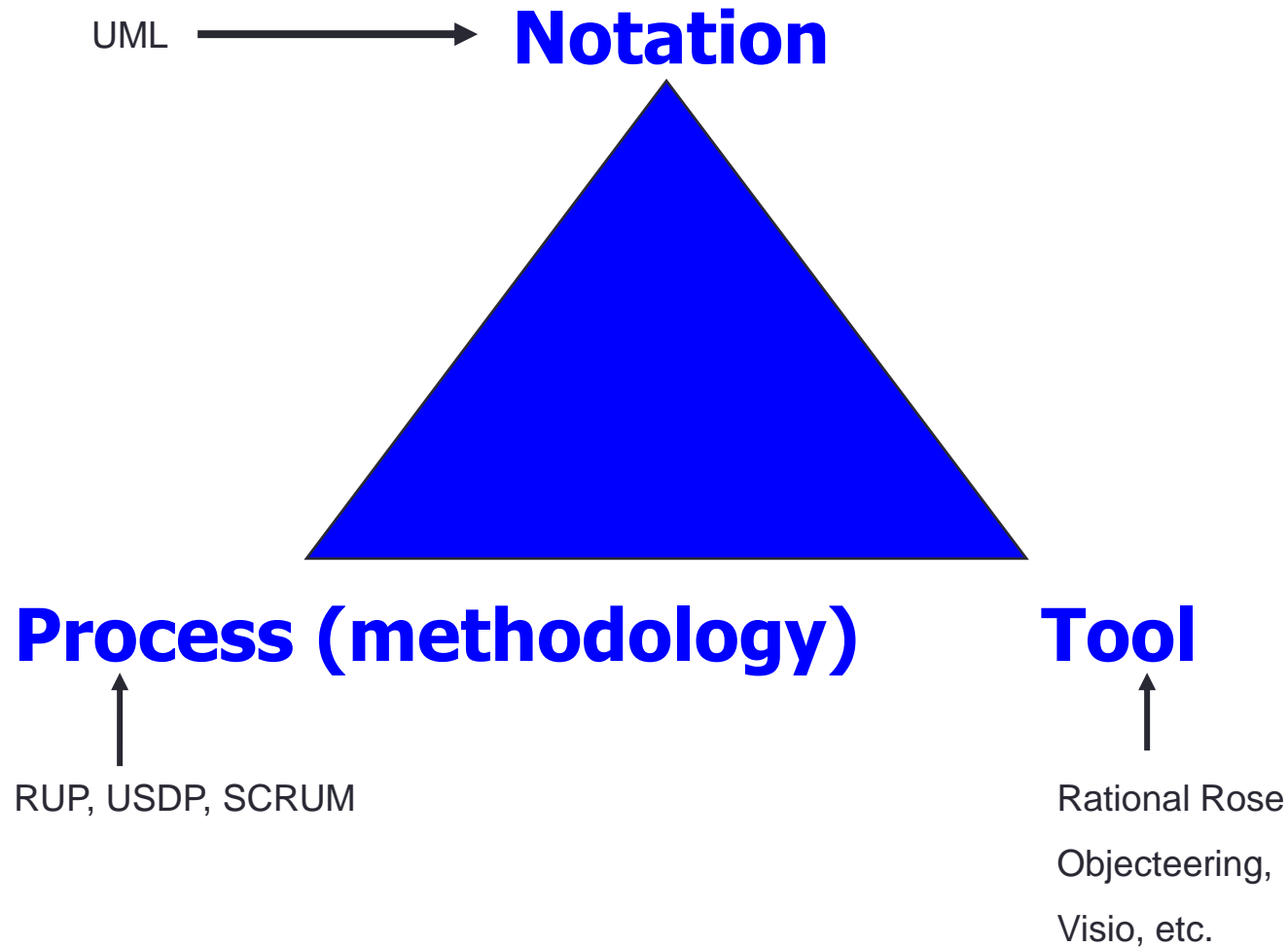
# UML - Evolution



Before '95' - Fragmentation ▶ '95' - Unification ▶ '98' - Standardization ▶ '99' - Industrialization



# UML: the success triangle



# UML

- **UML is not a method**, it is a notation to describe systems.
- Processes based on UML:
  - USD “Unified Software Development Process” by I. Jacobson.
  - RUP “Rational Unified Process” by Rational Software.
  - C. Larman “UML and patterns”.

# UML Charts

- **Class Diagrams (Part 1)**
- **Use Cases Diagrams (Part 2)**
- **Behavior Charts**
  - States Chart
  - Activity Chart
  - Interaction Diagrams
    - Sequence Diagram
    - Collaboration Diagram
- **Implementation Diagrams**
  - Components Diagram
  - Deployment Diagram

