Unified Modeling Language (UML)

An Overview

UML

- UML is a modeling ***notation***
  - standardized by OMG (proposal 1997, ver.1.1 in 1998, ver. 2.0 in 2004)
    - now in 2.5.1 (December 2017)
  - mature
  - based on
    - notations previously used in the software engineering OOA&D – Bracha .. , Booch (Ada at Rational), Rumbaugh OMT (at GE), Jacobson (use cases at Ericsson)
  - suitable for Object-oriented
  - design
  - implementation

UML Diagrams

- Defines a number of diagrams

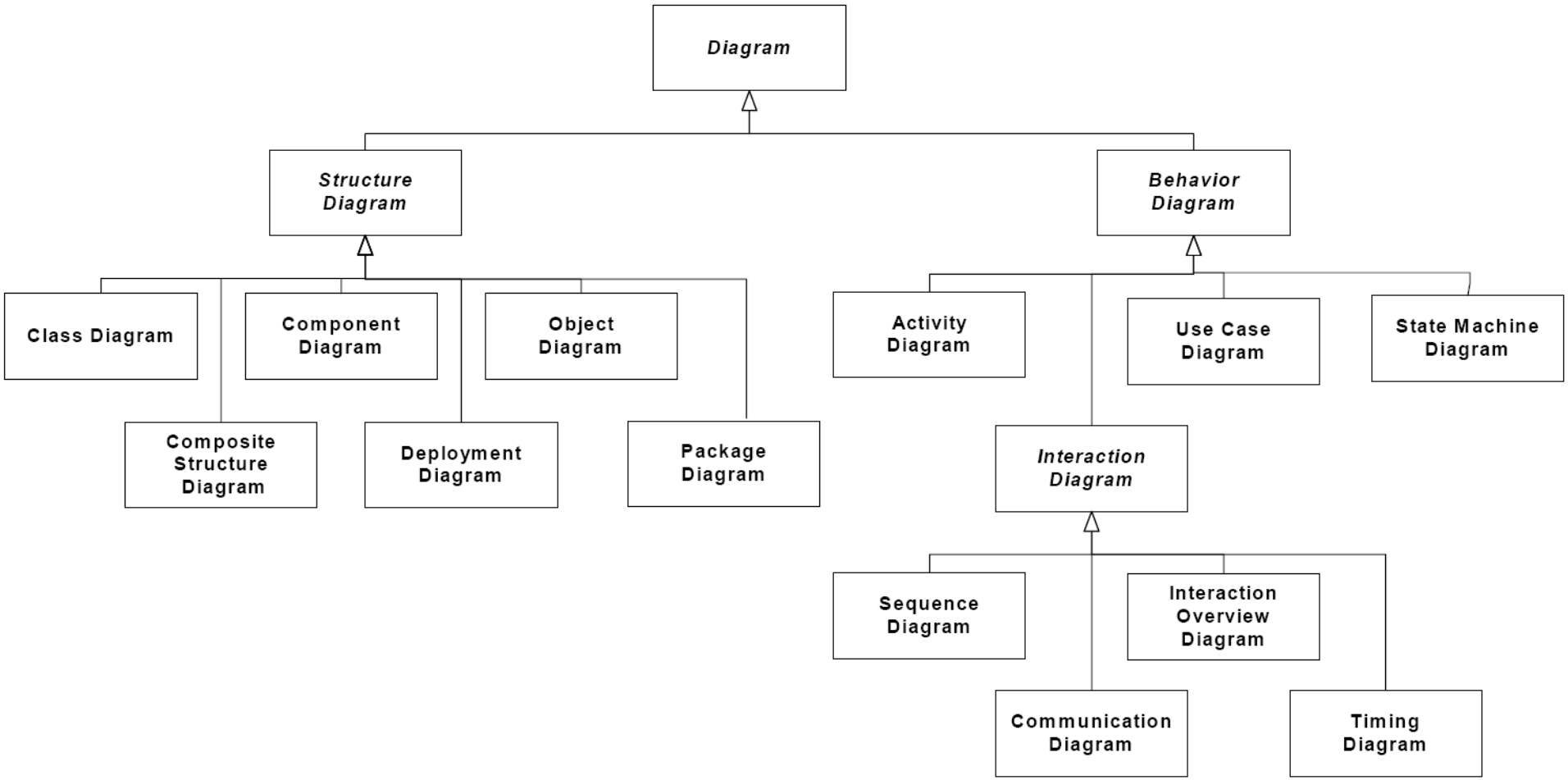
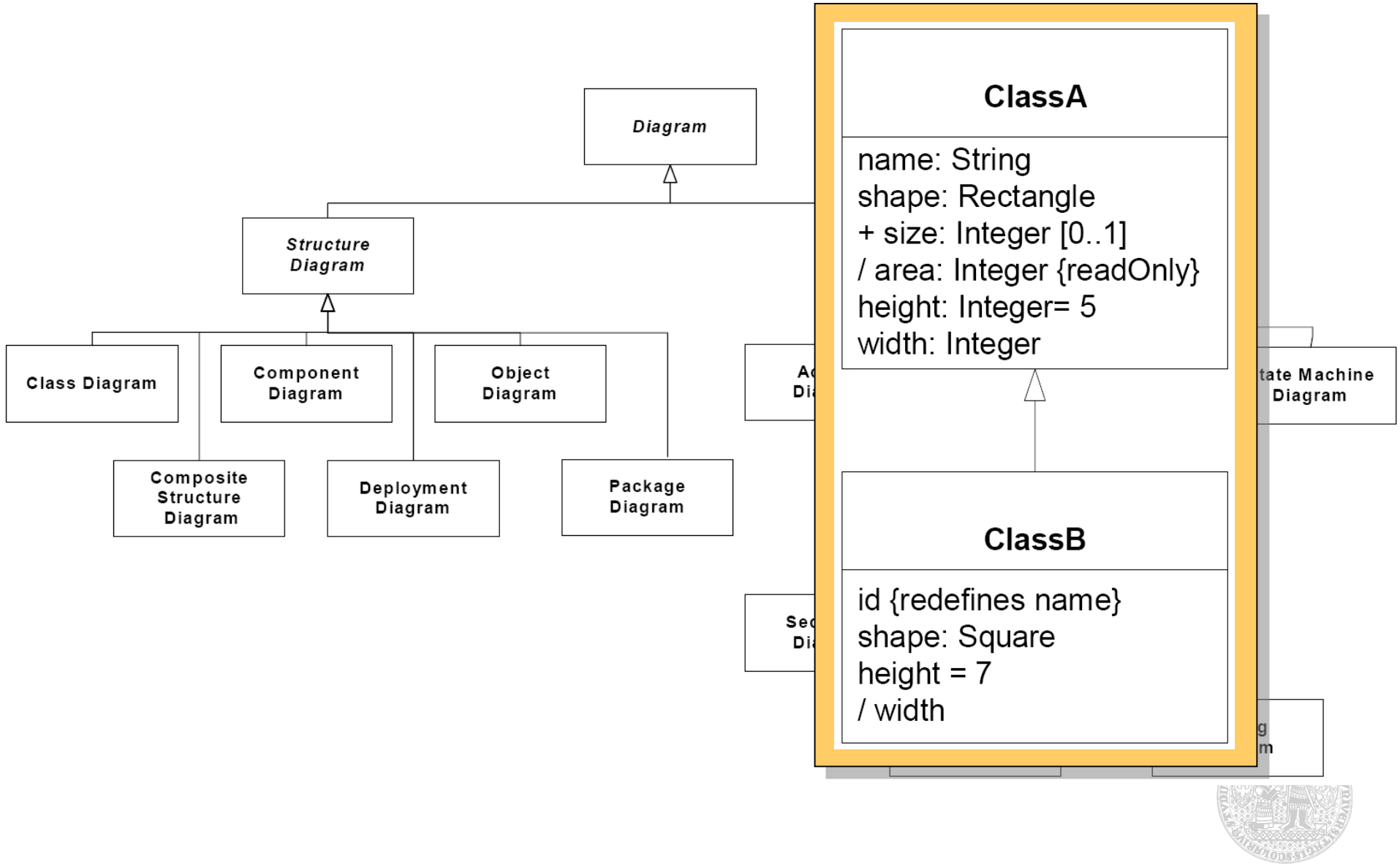


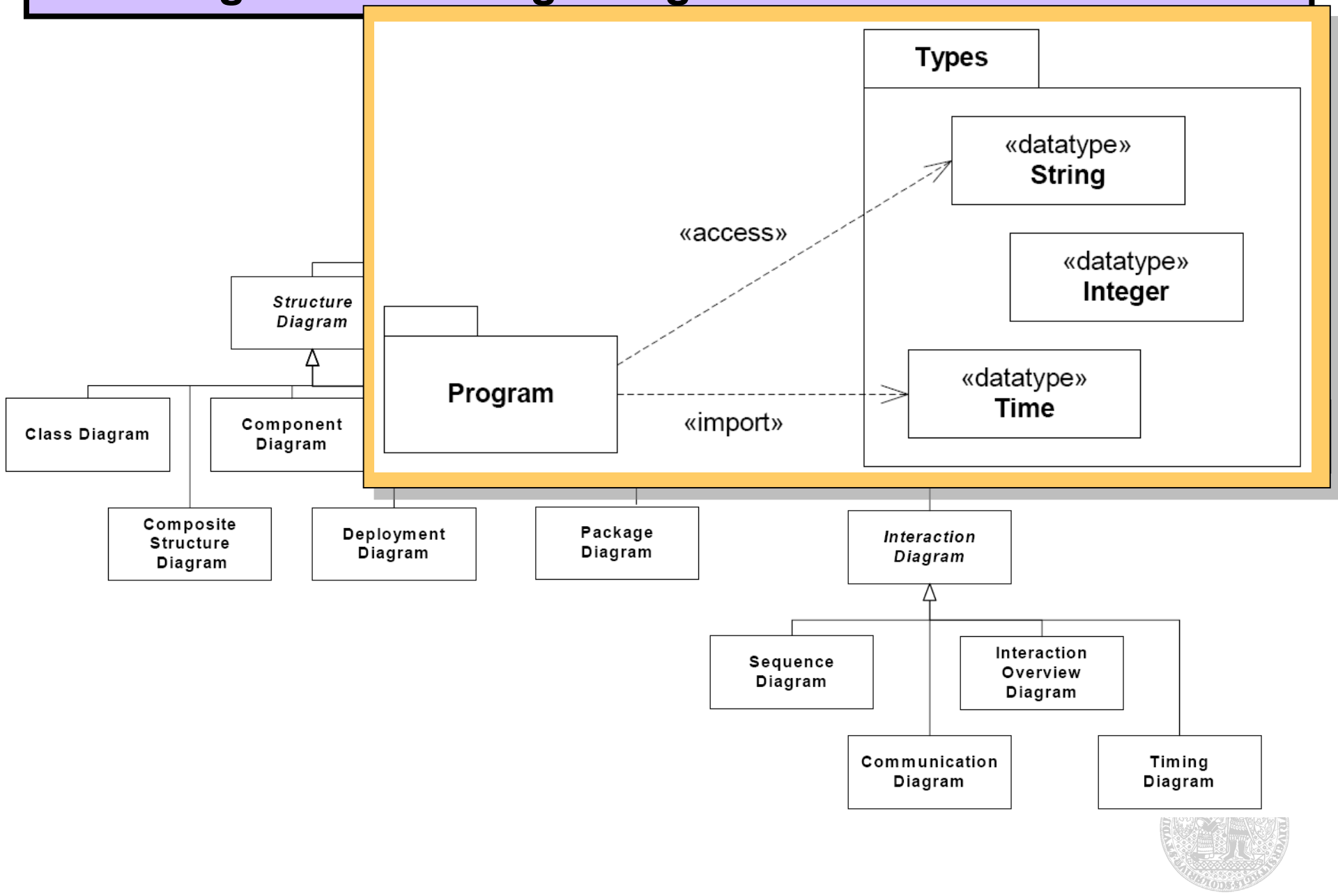
Figure from: OMG, "Unified Modeling Language: Superstructure, Version 2.1.1"

UML Diagrams – Class Diagram



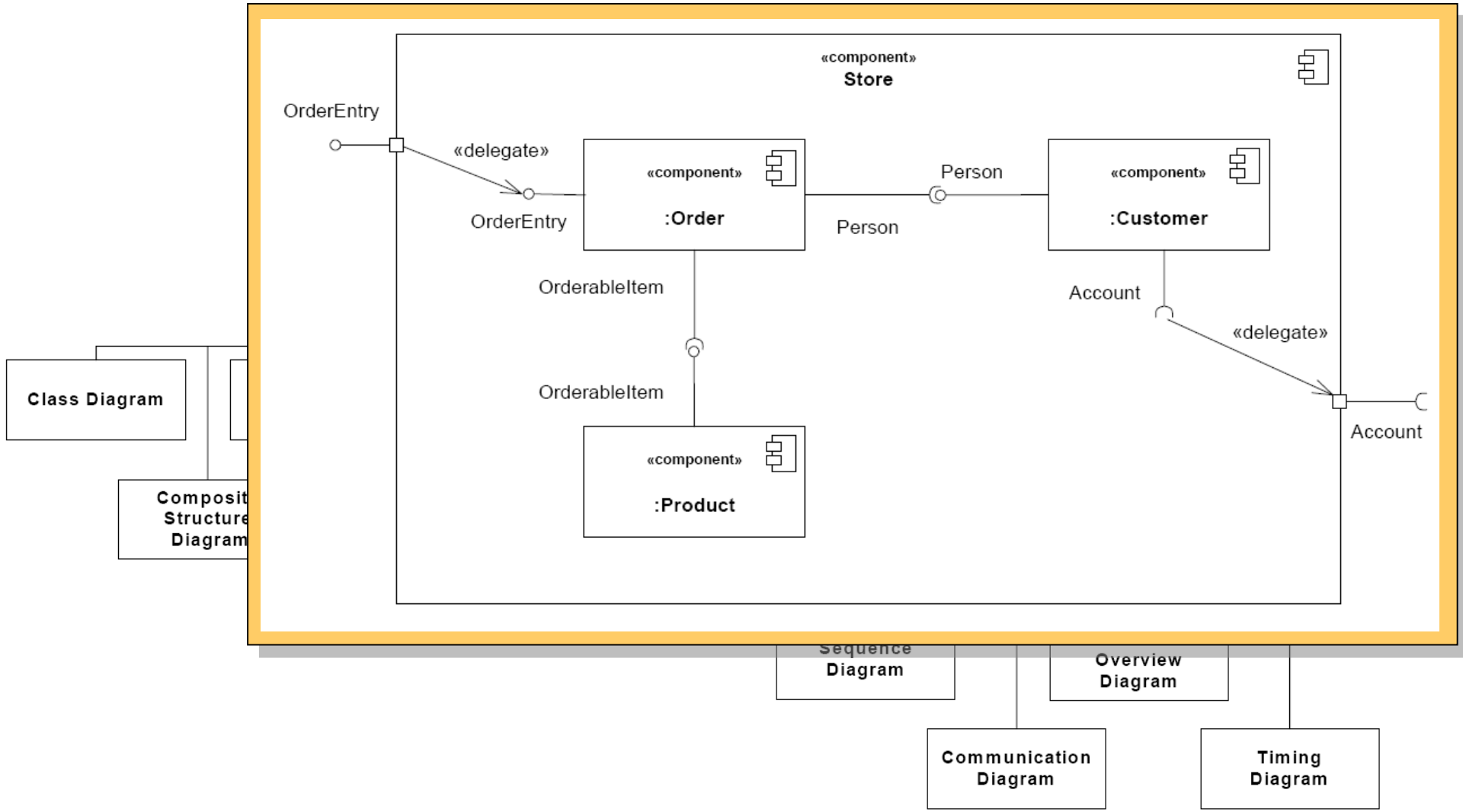
Figures from: OMG, "Unified Modeling Language: Superstructure, Version 2.1.1"

UML Diagrams – Package Diagram



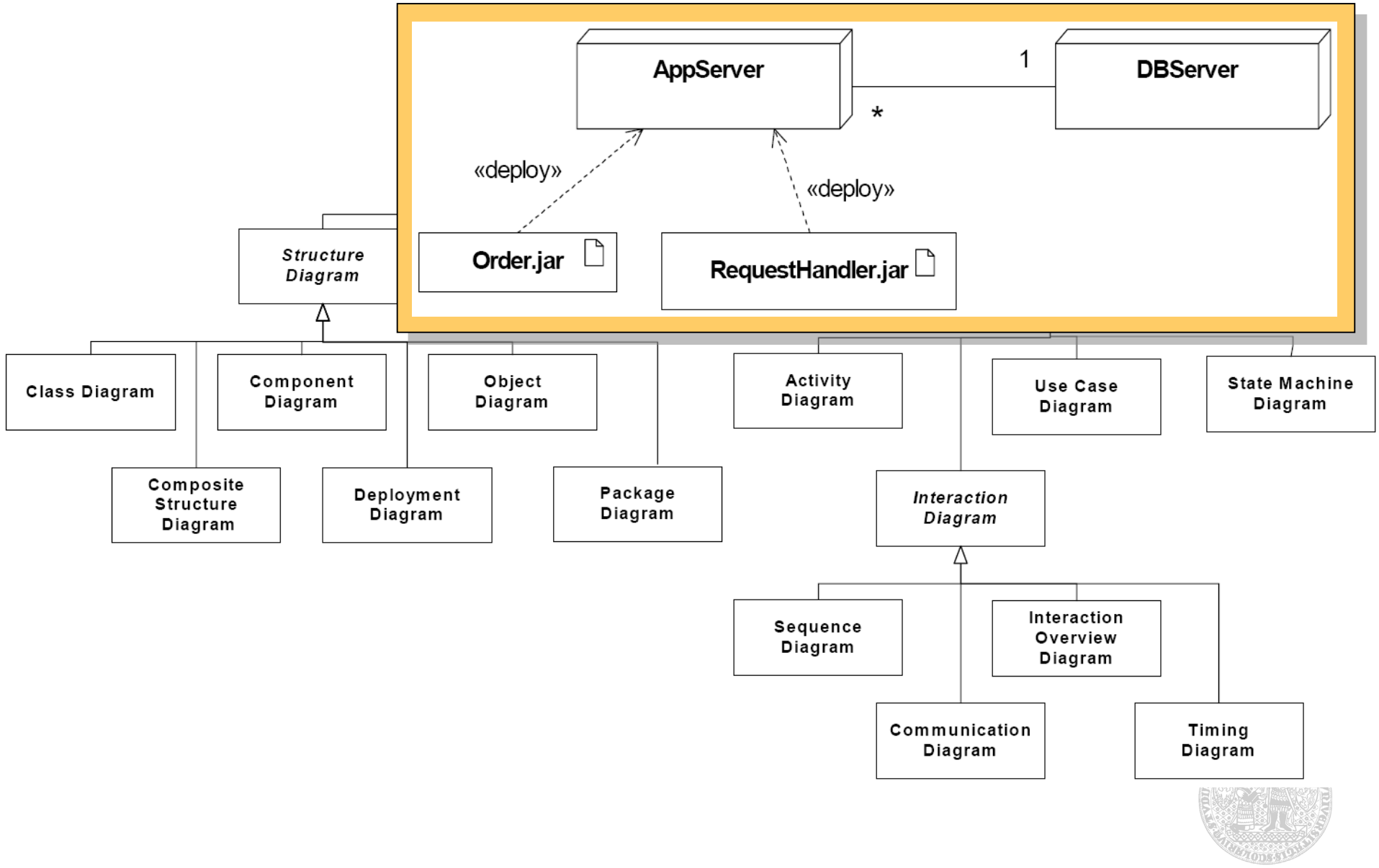
Figures from: OMG, "Unified Modeling Language: Superstructure, Version 2.1.1"

UML Diagrams – Component Diagram



Figures from: OMG, "Unified Modeling Language: Superstructure, Version 2.1.1"

UML Diagrams – Deployment Diagram



Figures from: OMG, "Unified Modeling Language: Superstructure, Version 2.1.1"

UML Diagrams

- Defines a number of diagrams

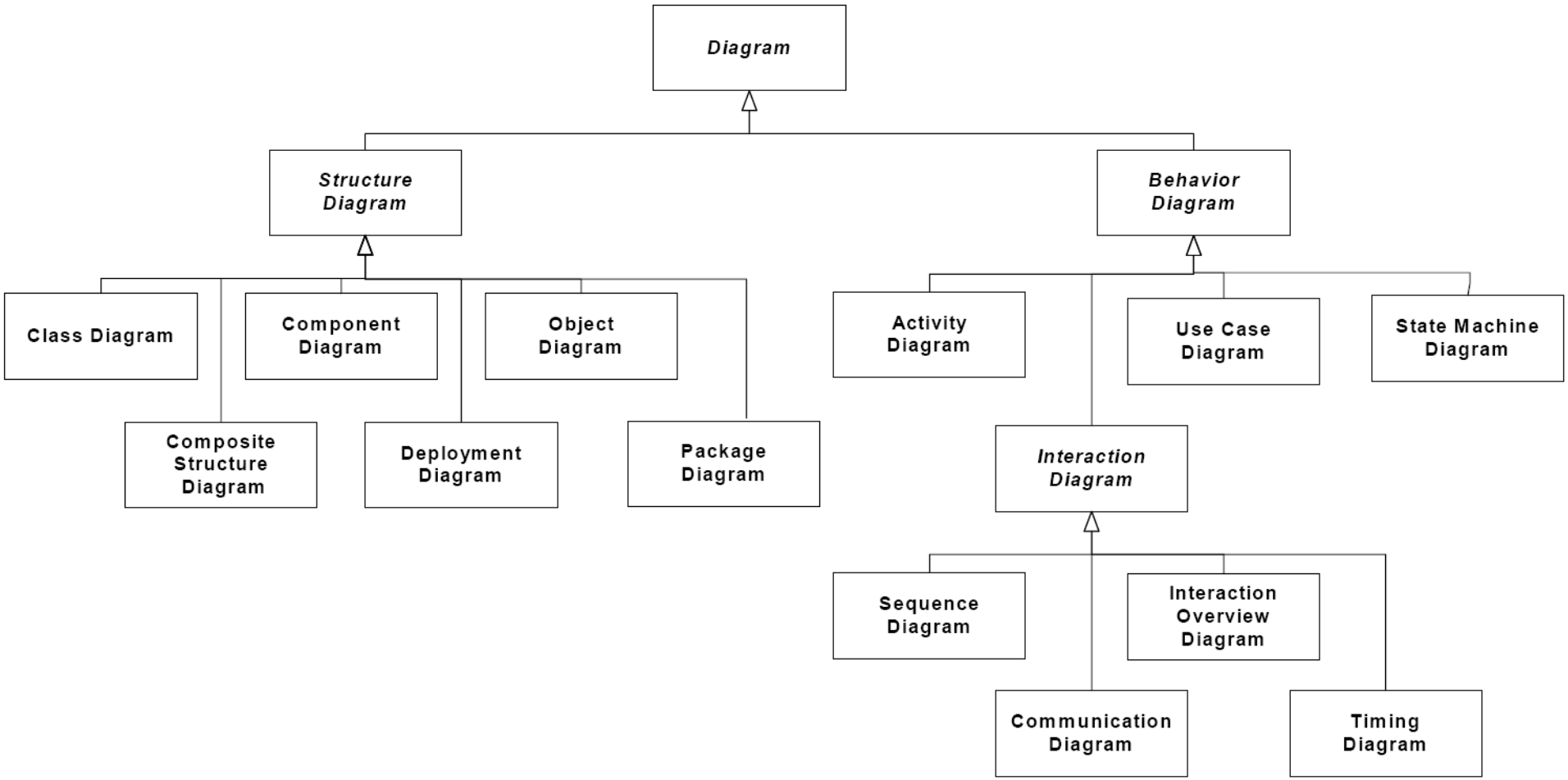
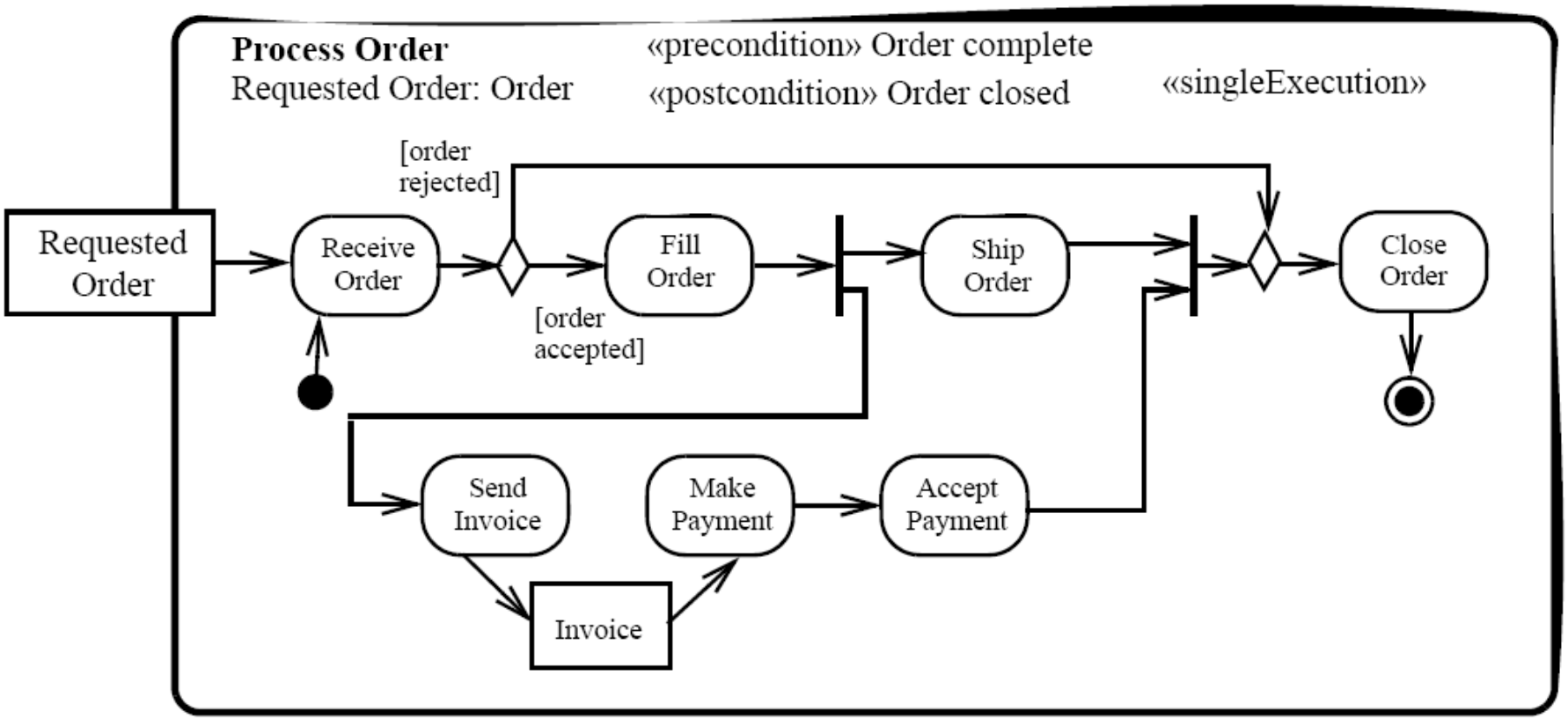


Figure from: OMG, "Unified Modeling Language: Superstructure, Version 2.1.1"

UML Diagrams – Activity Diagram

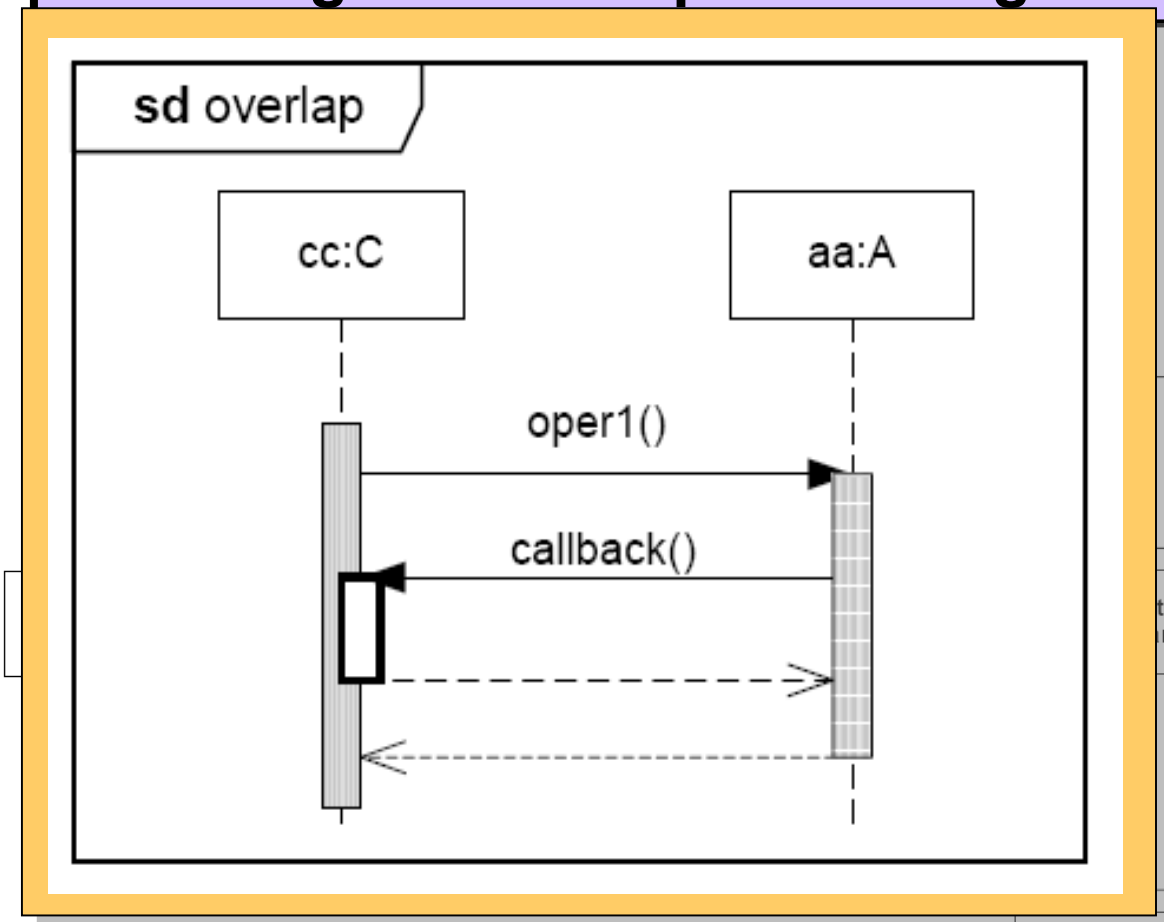


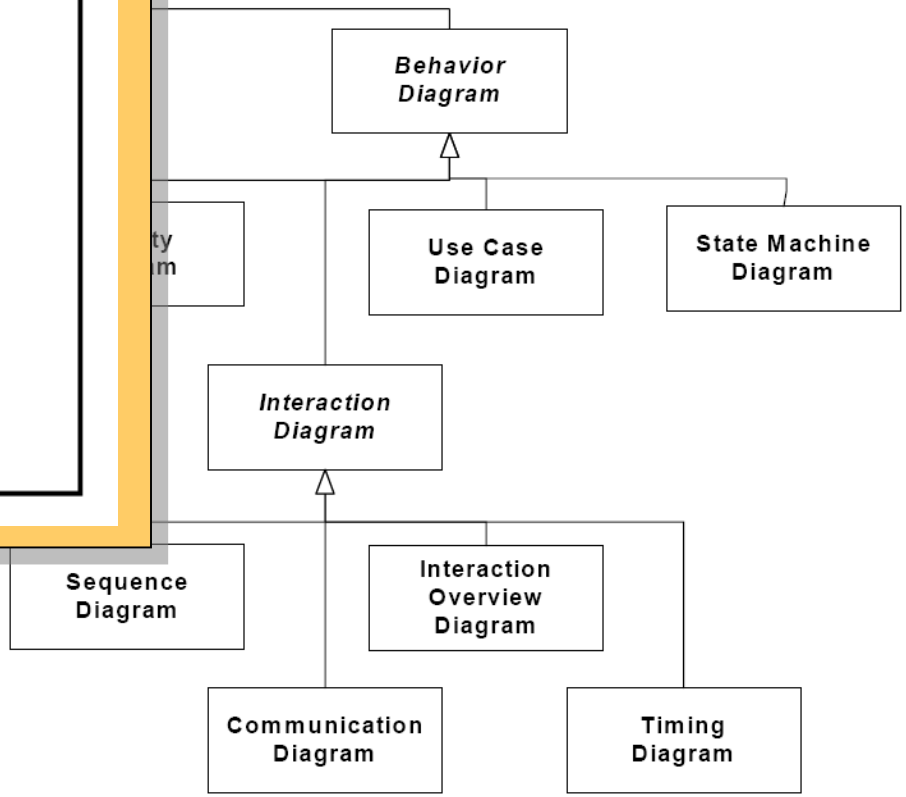
Communication Diagram

Timing Diagram

Figures from: OMG, "Unified Modeling Language: Superstructure, Version 2.1.1"

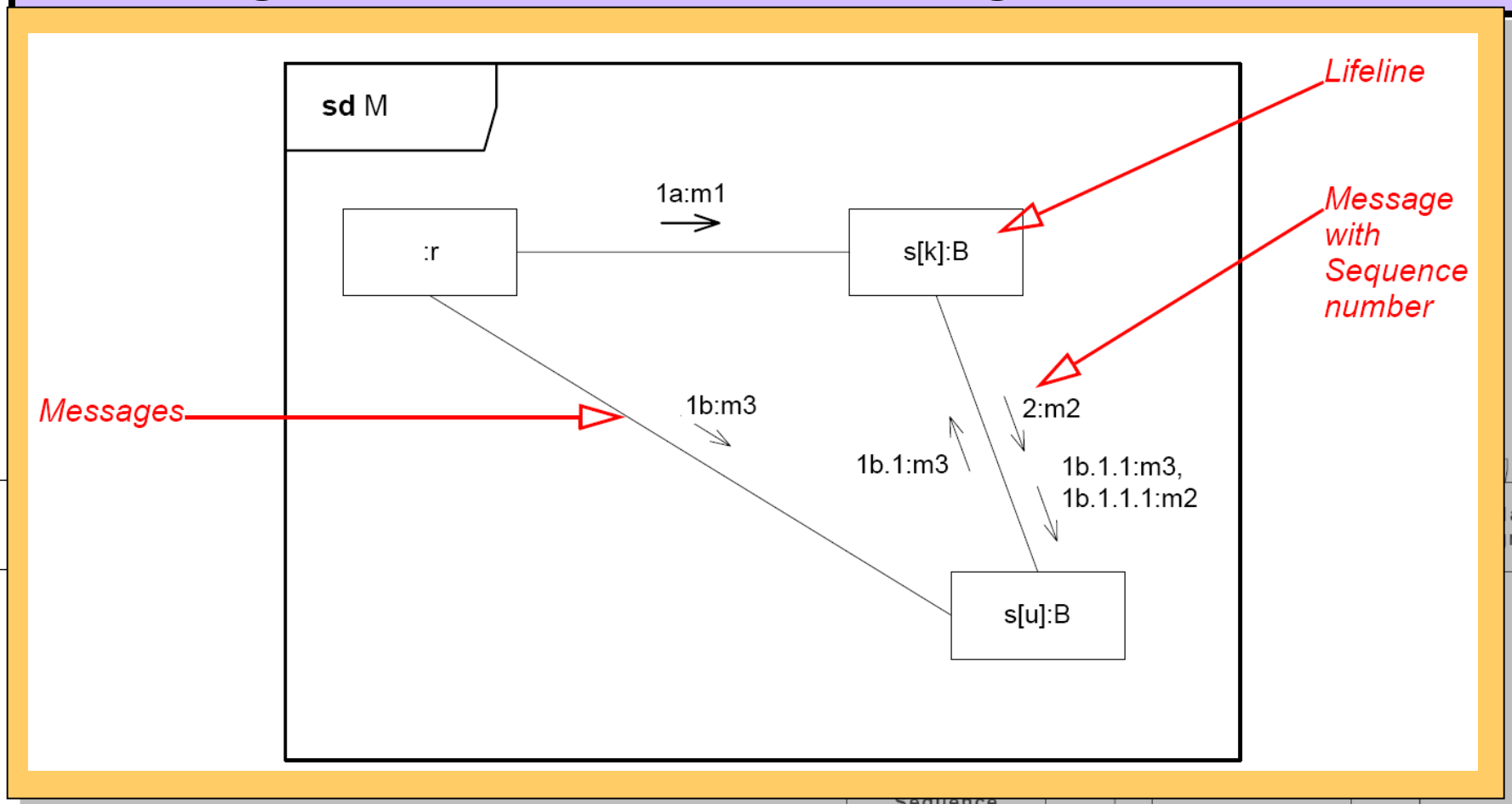
UML Diagrams – Sequence Diagram

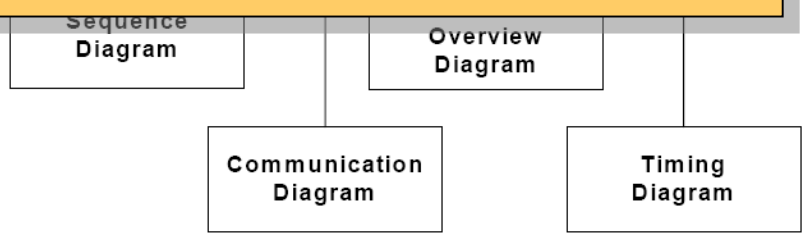




Figures from: OMG, "Unified Modeling Language: Superstructure, Version 2.1.1"

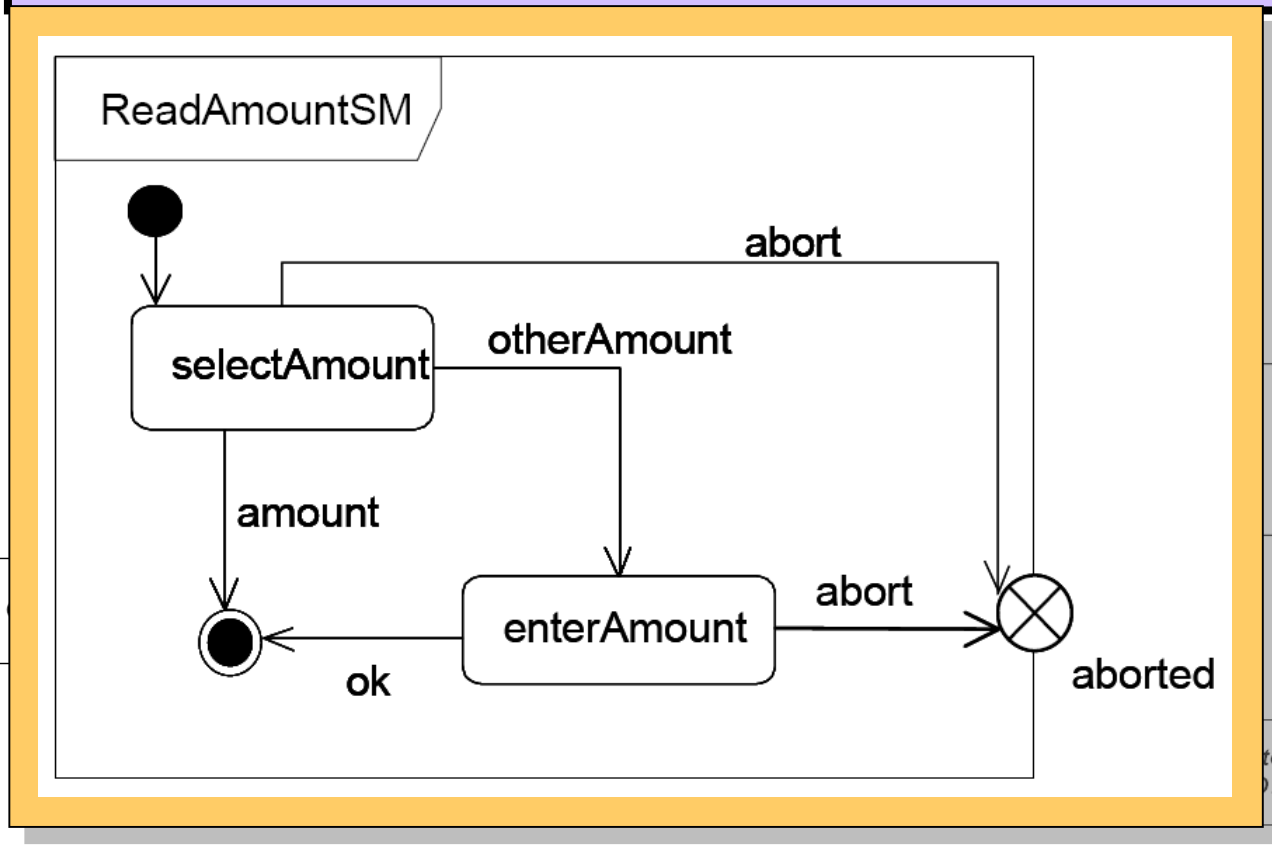
UML Diagrams – Communication Diagram

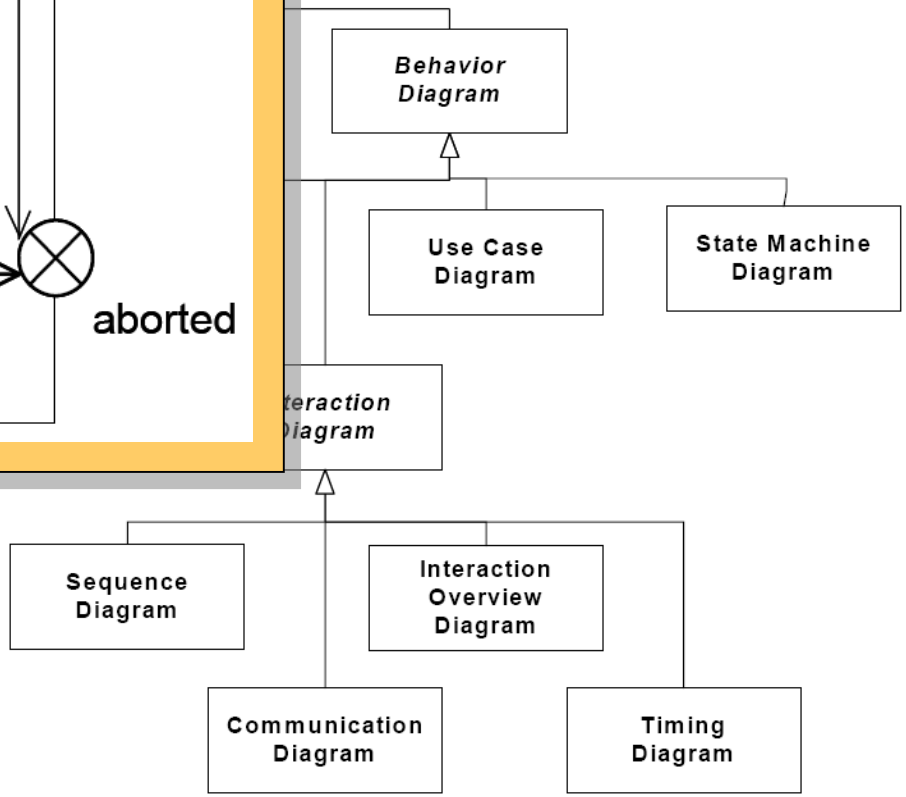




Figures from: OMG, "Unified Modeling Language: Superstructure, Version 2.1.1"

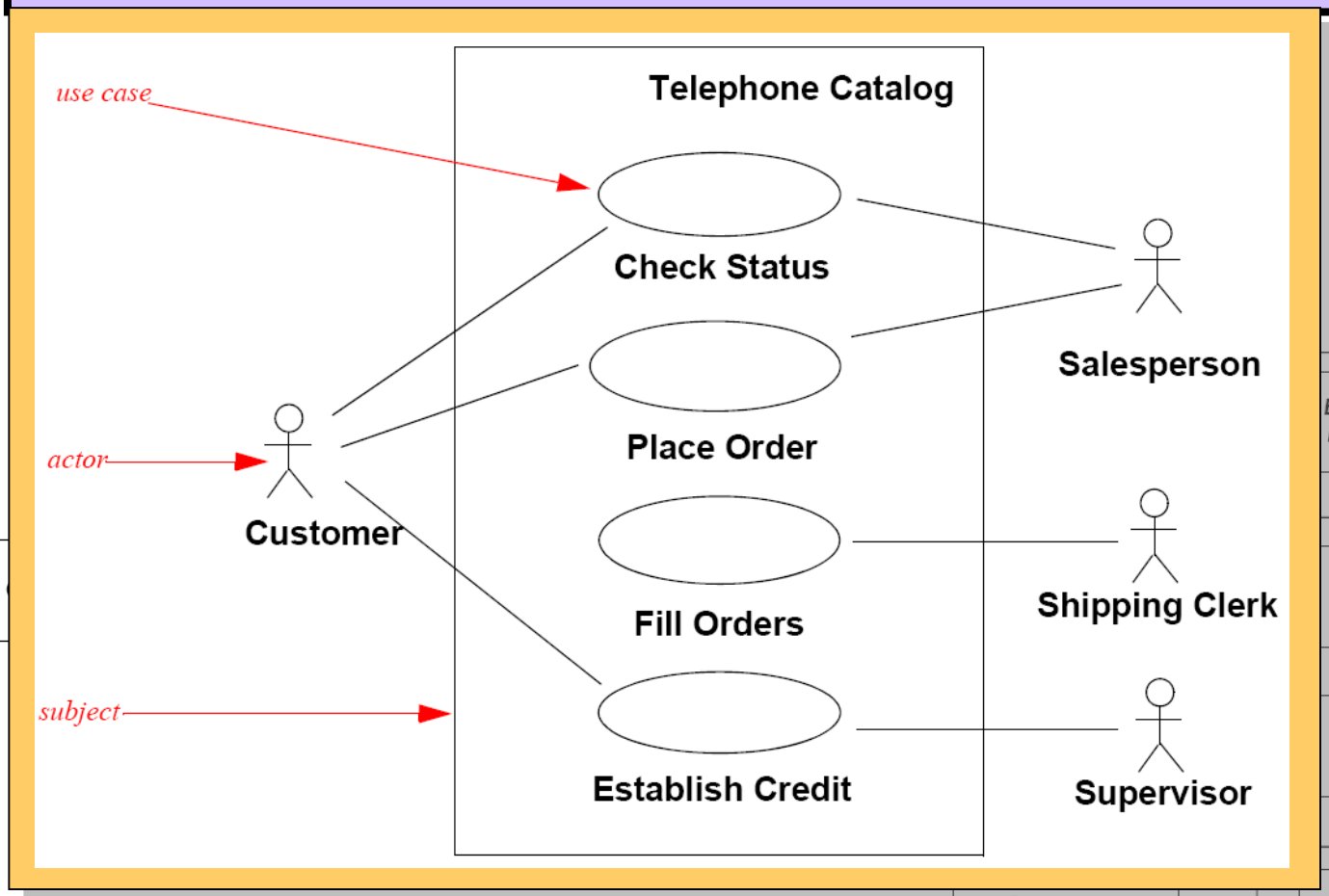
UML Diagrams – State Machine Diagram





Figures from: OMG, "Unified Modeling Language: Superstructure, Version 2.1.1"

UML Diagrams – Use Case Diagram



Figures from: OMG, "Unified Modeling Language: Superstructure, Version 2.1.1"

UML in Action

- CoCoME trading system...

Unified Modeling Language (UML)

Class Diagrams

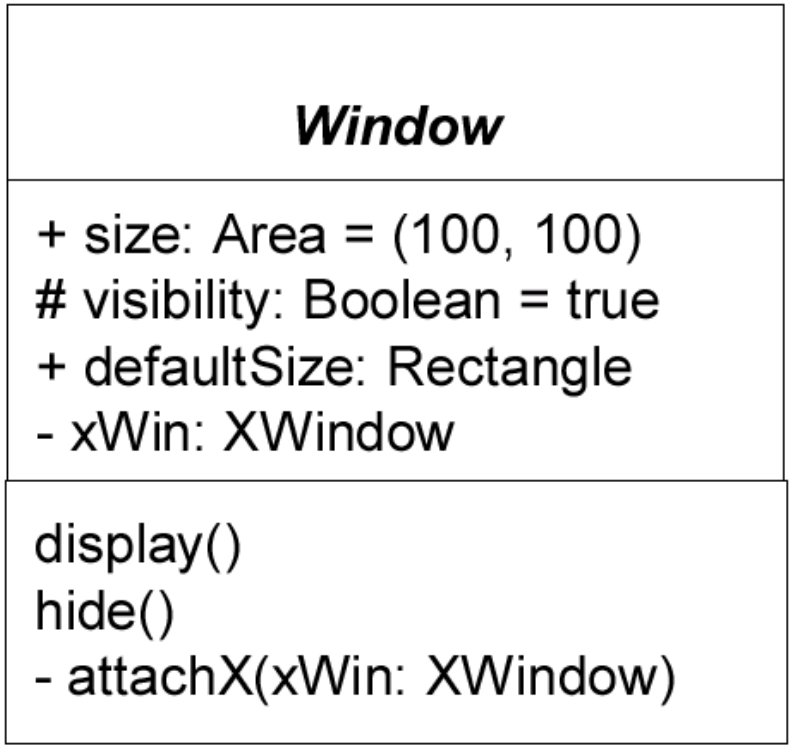
Class Diagrams

- A class diagram shows
  - classes
  - relations among classes
  - generalization
  - associations (with multiplicities, names)
    - special case: aggregation and composition

- It's definitely good to be able to read class diagrams!

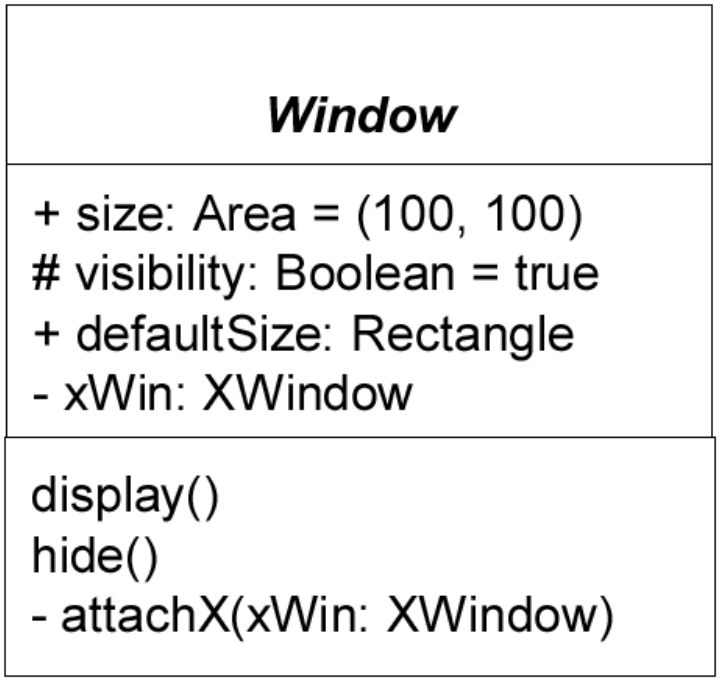
Class Diagrams – Class

- Shows a class with attributes with explicitly marked visibility



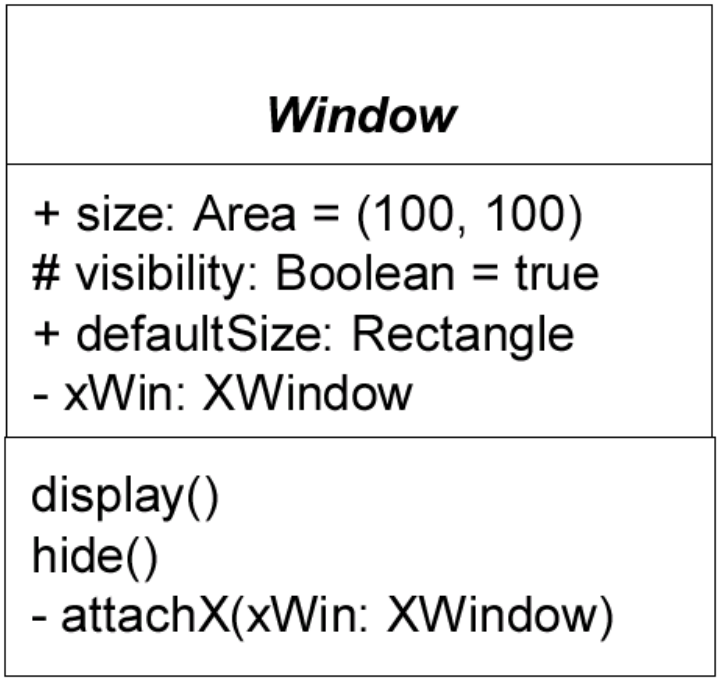
Class Diagrams – Notation

- Class compartments
  - top
    - name and annotations
    - stereotypes, superclass,...
  - attributes
  - operations
  - additional compartments
    - added by extensions, e.g.
    - EJB finder/business/activation compartments...



Class Diagrams – Notation

- class name: bold
- abstract class (or method): italics
- class scope (aka static): underlined (instance-scope otherwise)
- visibility (attributes, operations)
  - + public visibility
  - # protected visibility
  - - private visibility
  - ~ package visibility



Class Diagrams – Example

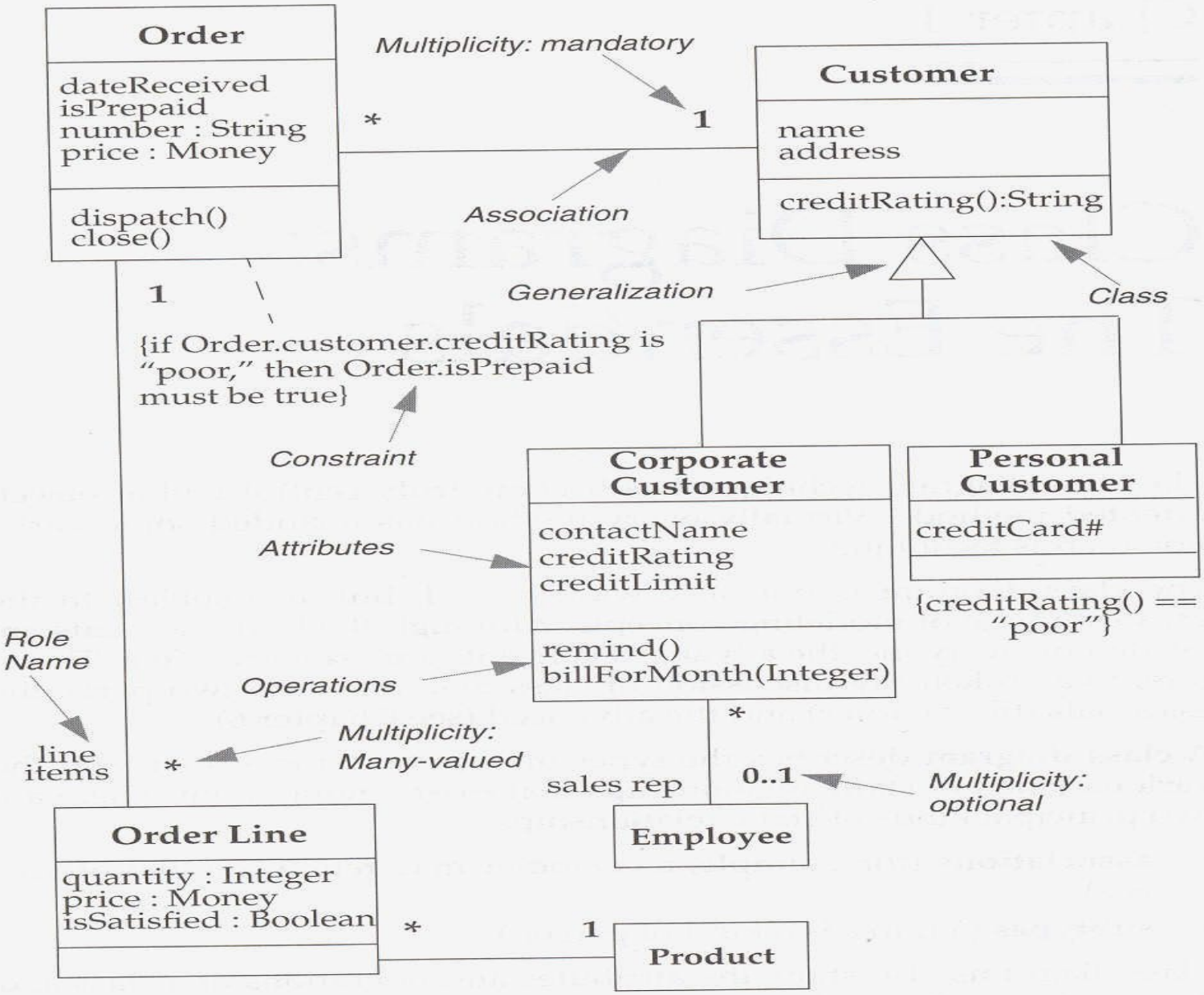


Figure from: Larman, C., "Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process"

Class Diagrams – Example

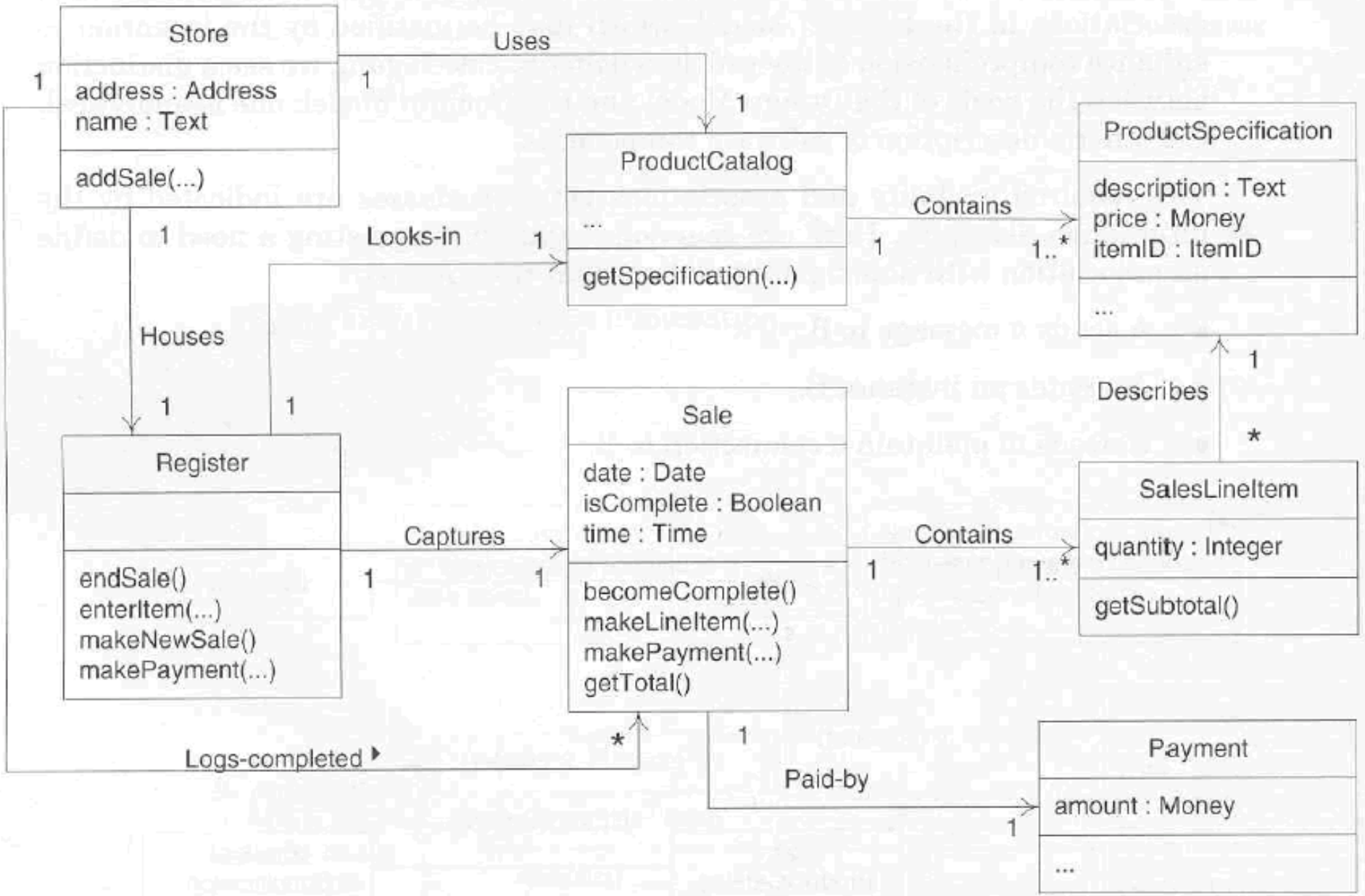
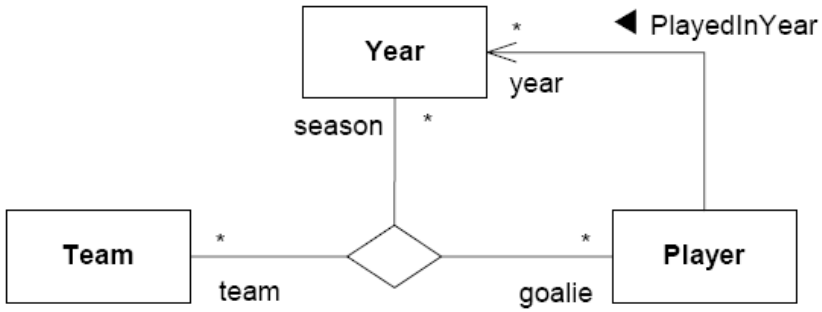
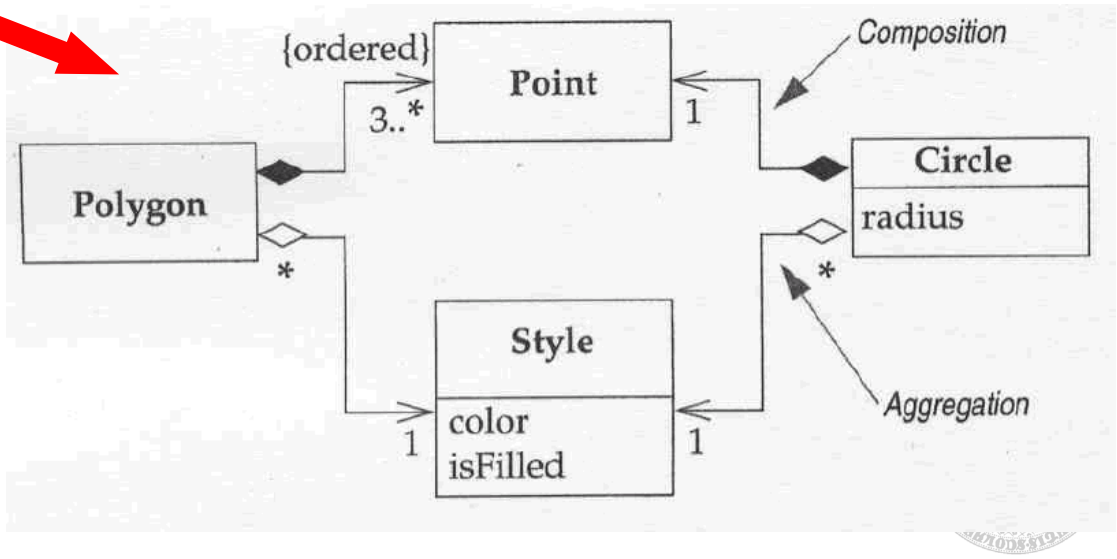


Figure from: Larman, C., "Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process"

Associations

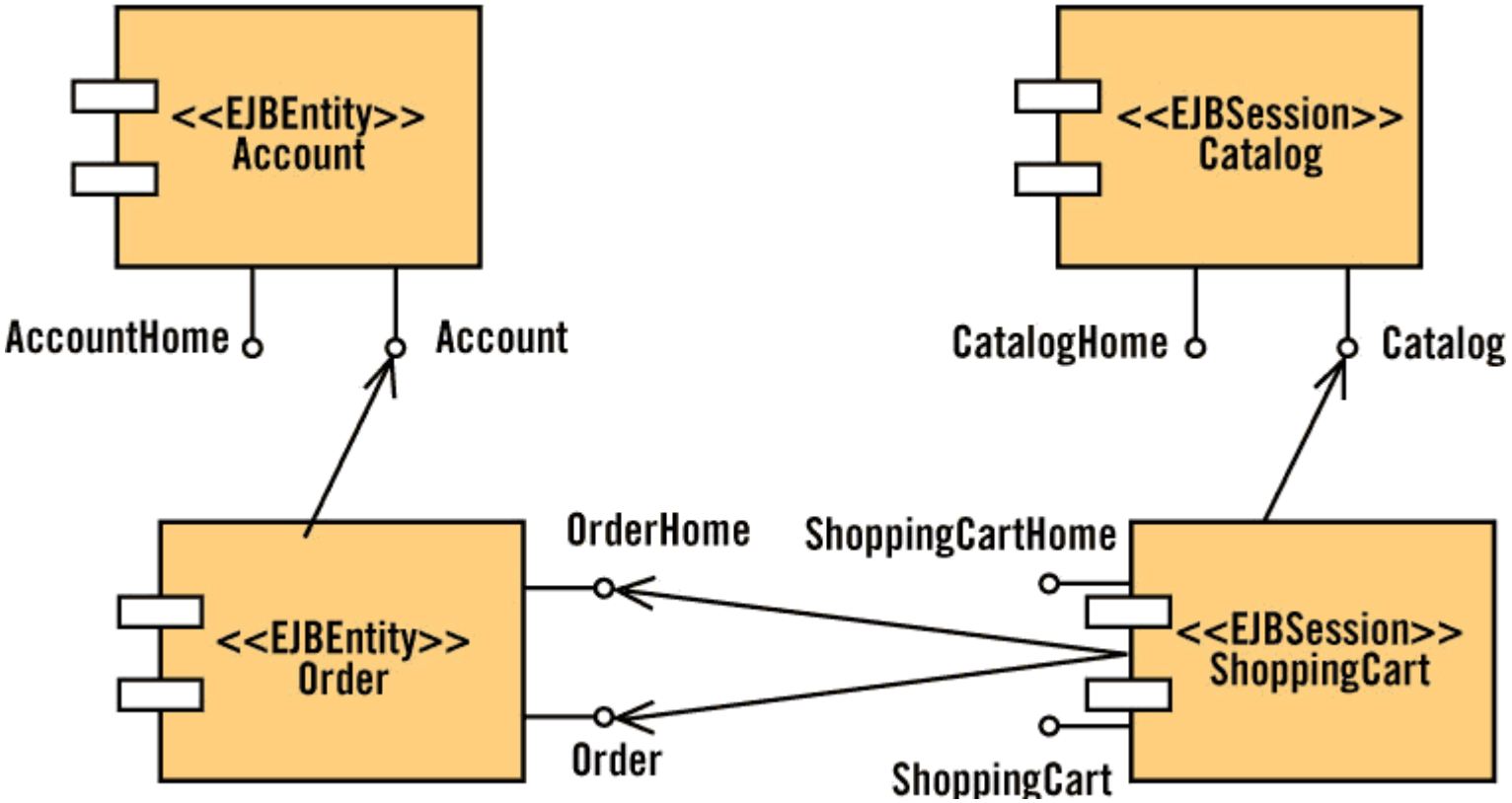
- Association
  - Semantic relationship
  - At least two ends
    - May be navigable
    - May have assigned aggregation type
      - Shared
      - Composite





Profiles & stereotypes

- UML can be extended by defining so called profiles and stereotypes
- This allows assigning particular roles and associating additional attributes to existing UML blocks

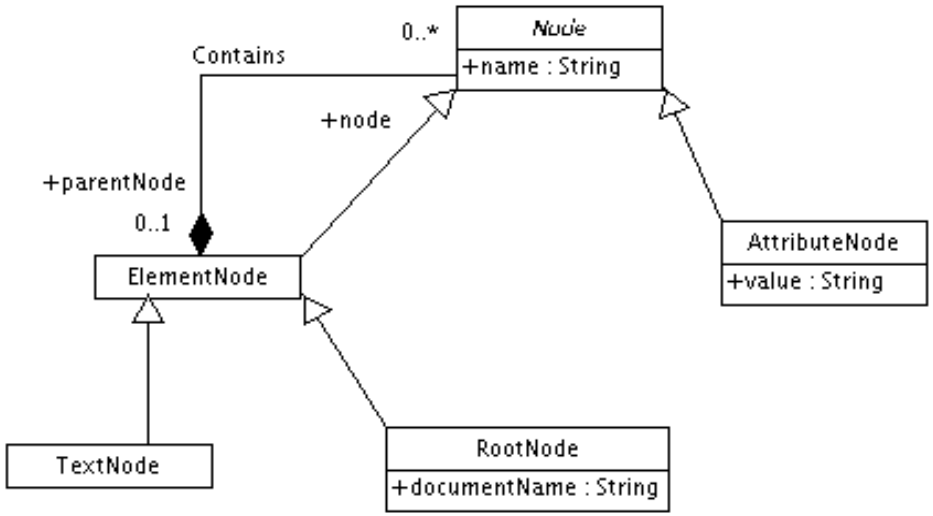


Meta-models

Modeling models...

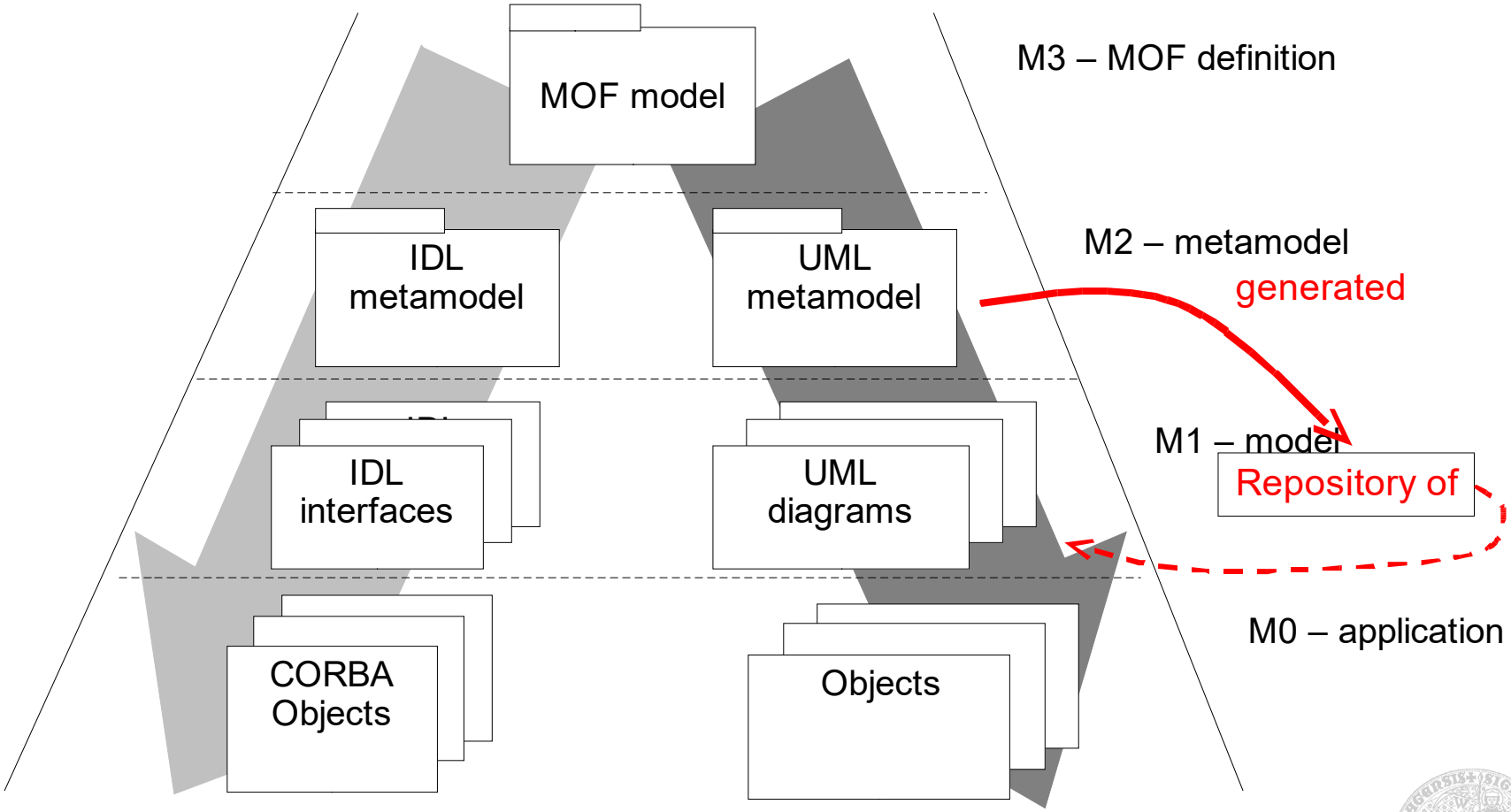
Models and meta-models

- Model is what we specify
  - e.g. data model of an application modeled in UML
- But what the language, which we use for modeling?
  - The language itself can be again described by a model
  - This model is called meta-model
- Meta-modeling constructs
  - Classes
  - Associations
  - DataTypes
  - Packages
  - Constraints

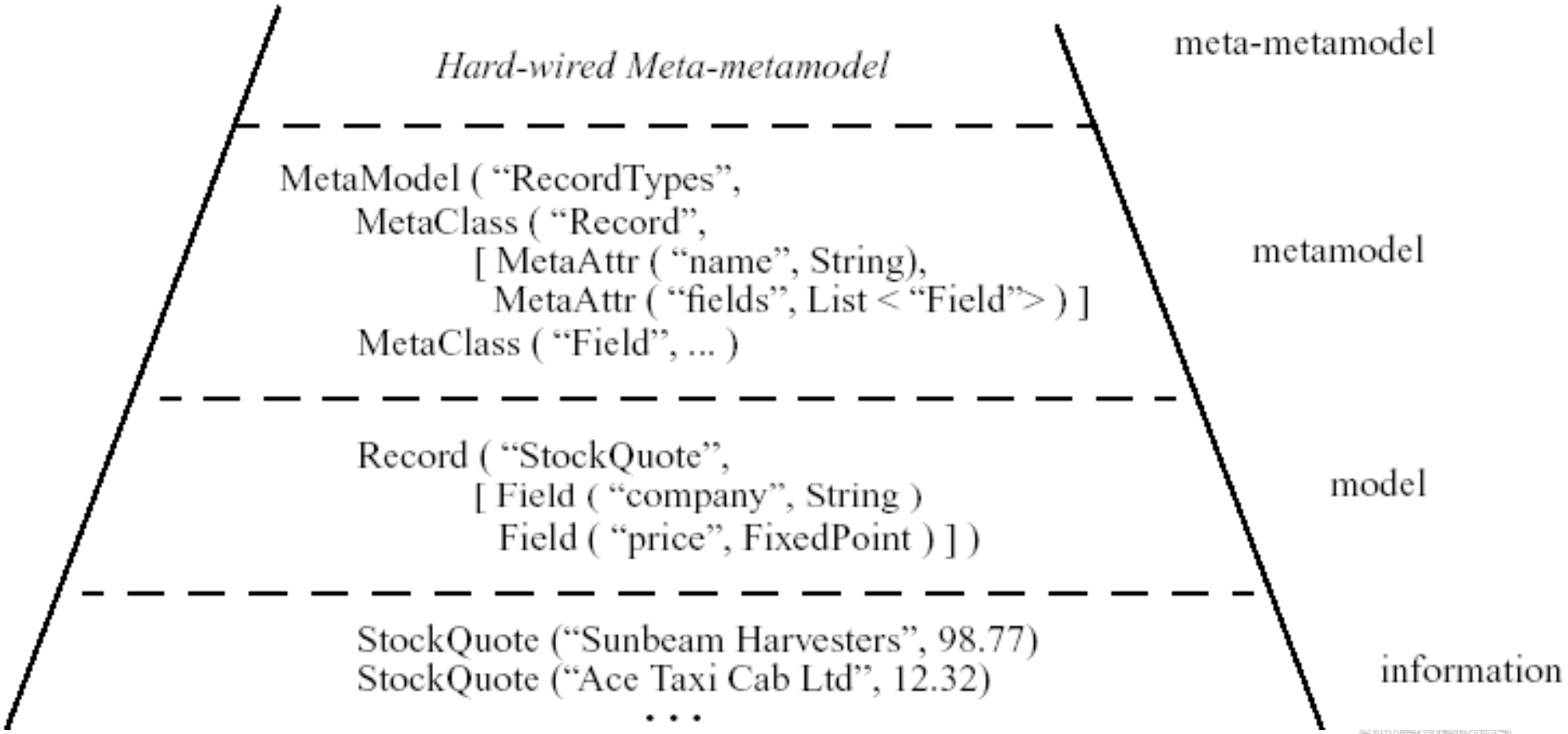


XML metamodel

Modeling hierarchy



Modeling hierarchy



How many meta layers?

- The minimal number of layers is two

- Examples
  - 2 layers
    - generic reflective systems - Class/Objecs
  - 3 layers
    - relational database systems - SysTable/Table/Row
  - 4 layers
    - UML, MOF specification - MOF/UML/User Model/User Object
      - MOF is a UML-like language for meta-modeling (i.e. only core constructs compared to UML)

Representing MOF

- MOF has no own graphical representation
  - Uses UML
  - Relies on the fact that UML and MOF have a lot of similarities

- Brain exercise:
  - UML is M2-model
    - Thus, it is an instance of MOF
  - UML is used to represent MOF models
  - MOF is modeled in MOF
    - Thus, MOF is formalized by UML

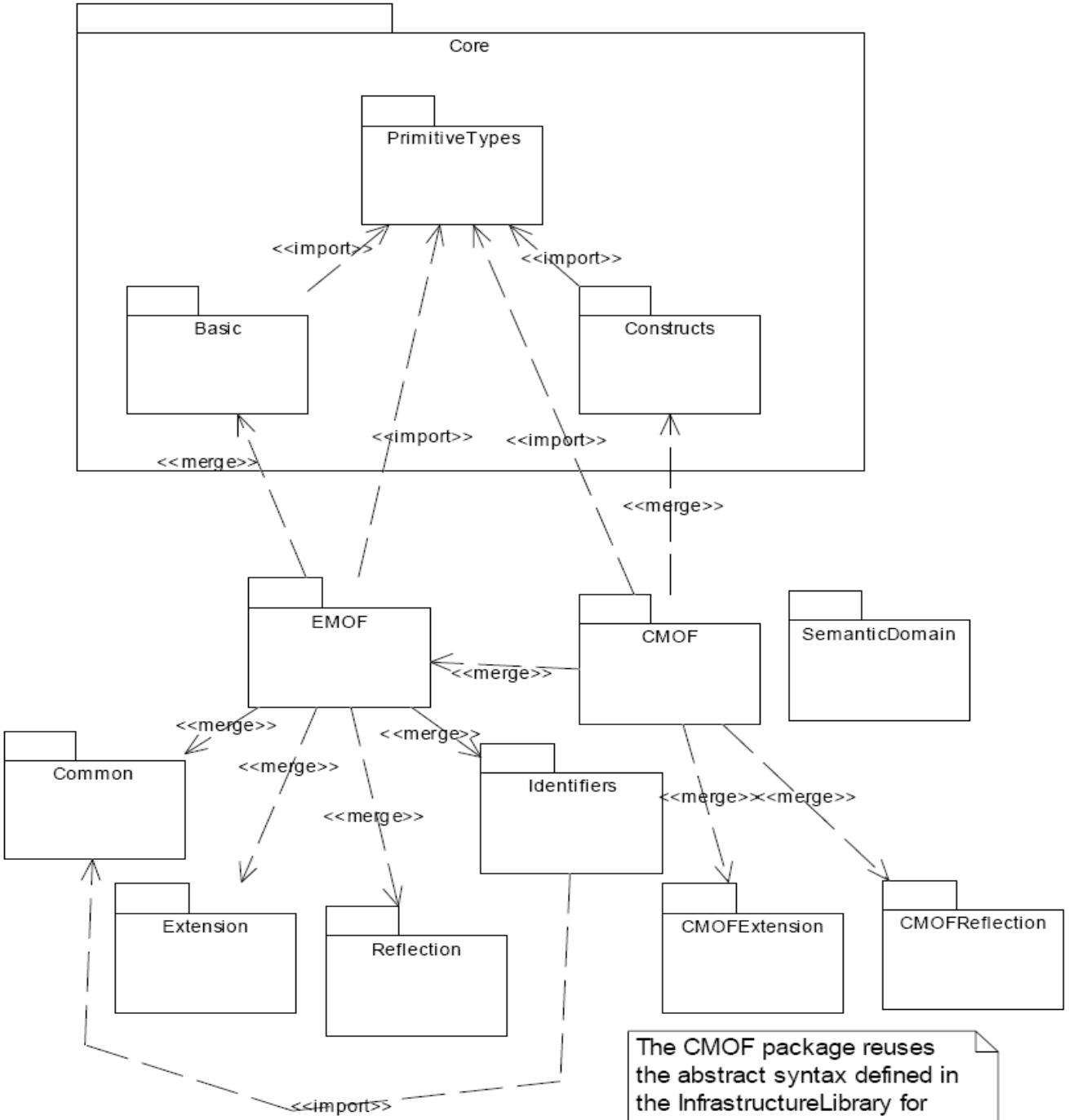


Figure from: OMG, "Meta Object Facility (MOF) Core Specification, Version 2.0"

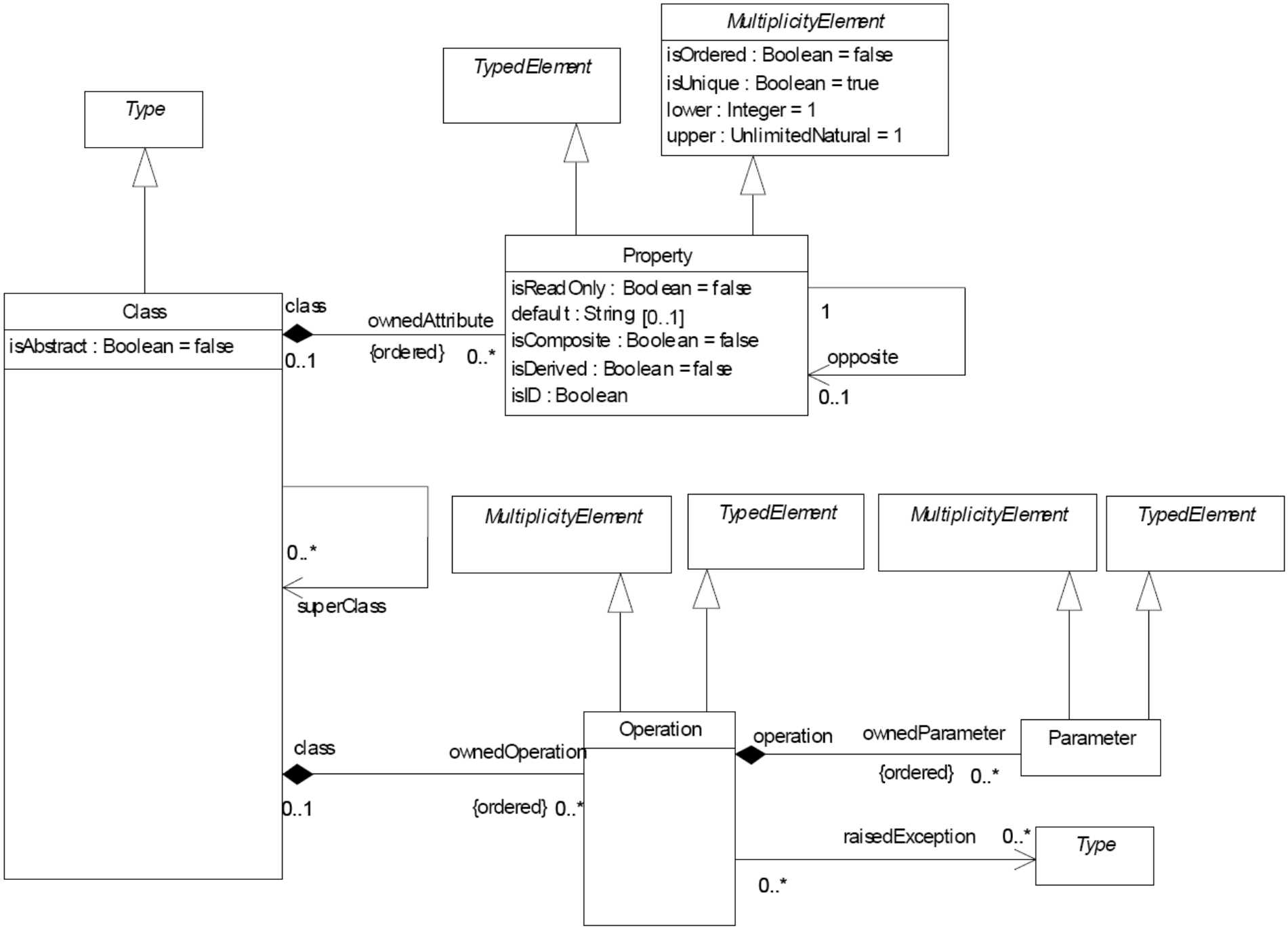


Figure from: OMG, “Meta Object Facility (MOF) Core Specification, Version 2.0”