Meta-data modeling

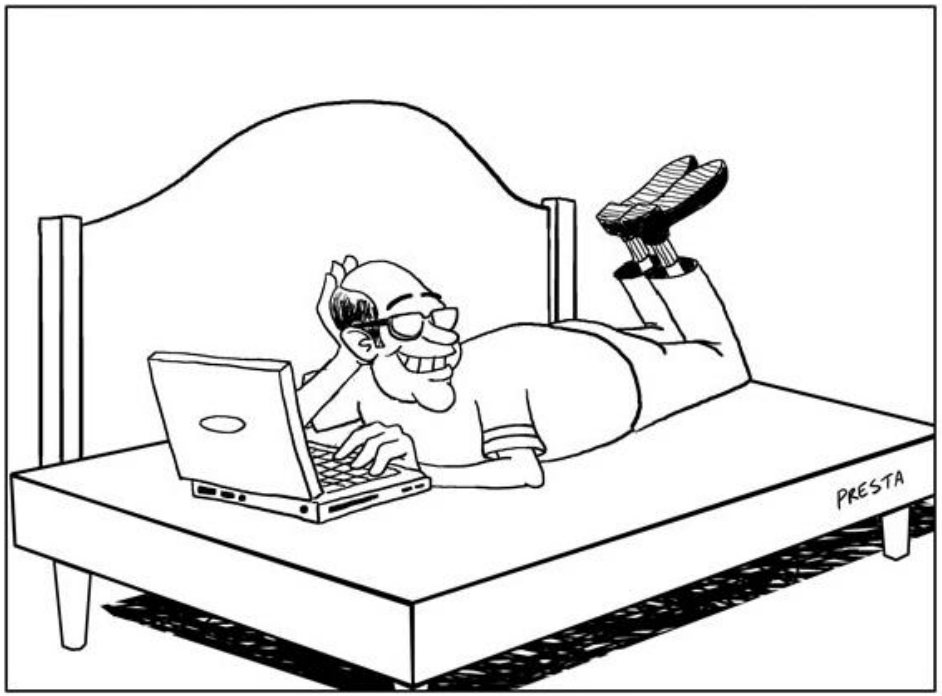
Session 3
12 February 2019

Prof.dr. Sjaak Brinkkemper
Dr. Sietse Overbeek

Universiteit Utrecht

Agenda

- Introduction to meta-data modeling
- PDD notation
- Examples and assignments

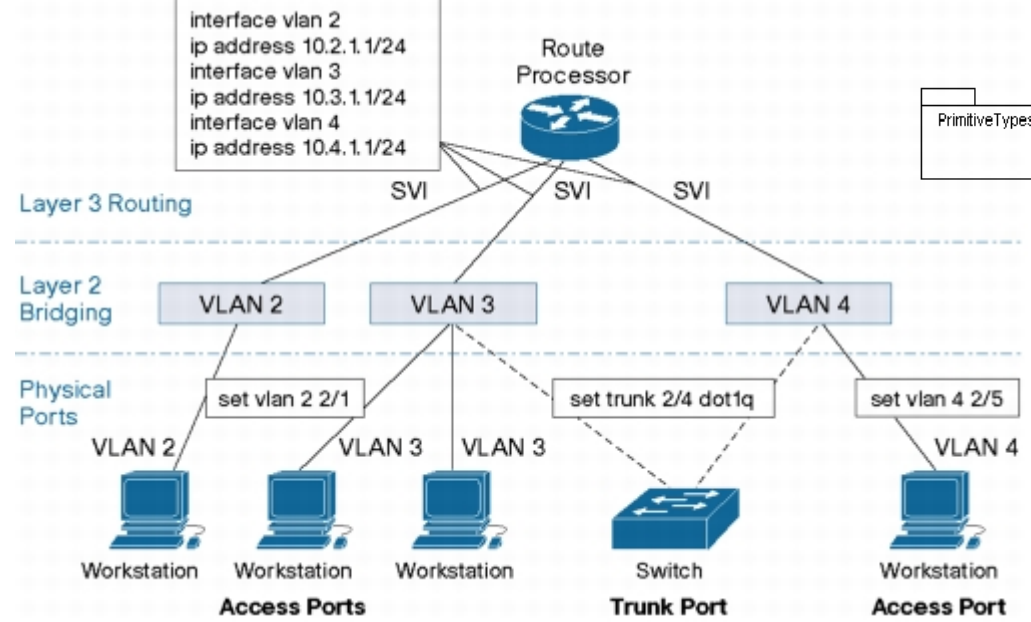


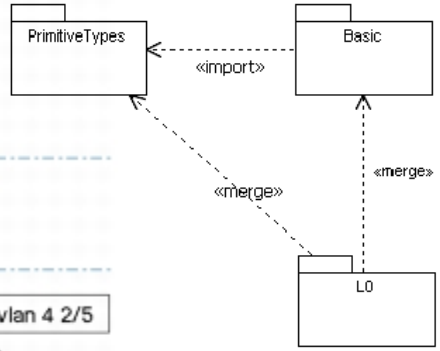
Data Modeling Centerfold

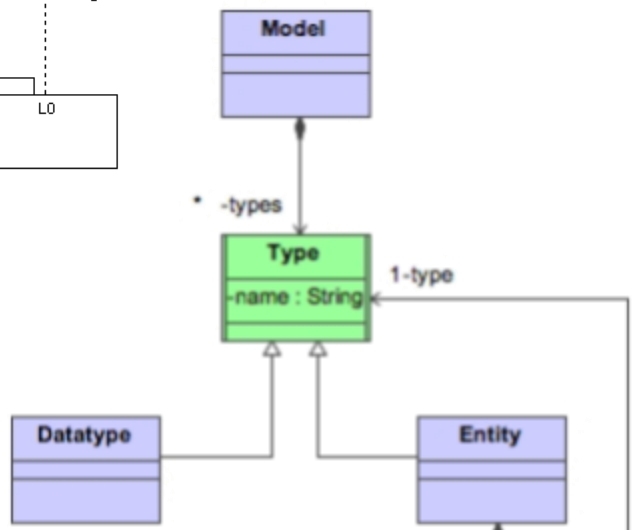
Recall from the first lecture:

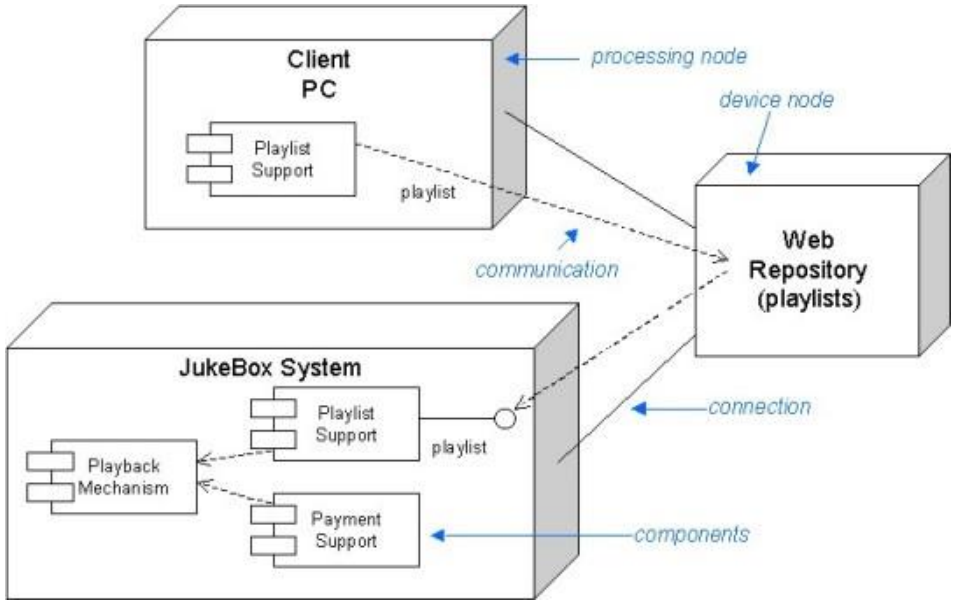
- A model is a representation that contains statements about the properties of an artifact (object) in a real or imagined world (universe of discourse).
- A meta-model is a model that consists of formal statements about models. It is yet another abstraction, highlighting properties of the model itself.
- Meta-models can be created for both processes and data. In this lecture, we focus on data.

Data-models on different levels









ROW	FIRST NAME	LAST NAME	AGE
#1	Bob	McBob	42
#2	John	Johnson	24
#3	Steve	Smith	38

(Meta-)meta-models

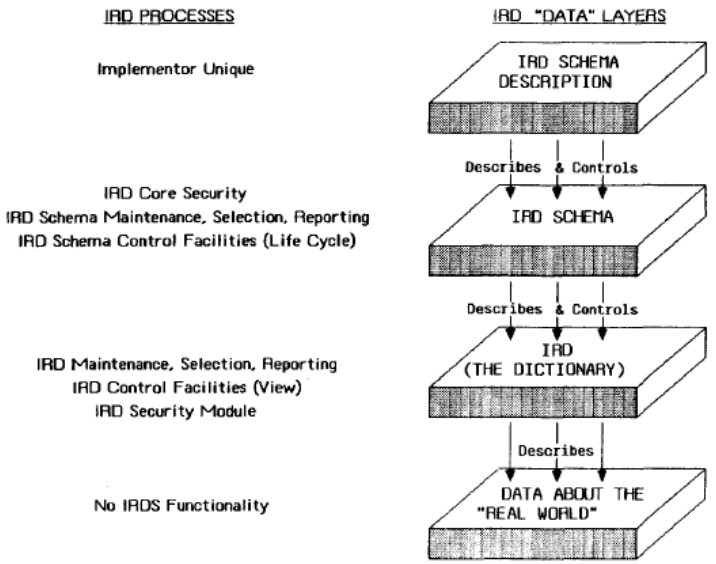
- Why do we need them?

- Meta-models can be used to formalize notations:
  - Meta-models are used to describe the grammar
  - A meta-model describes all models one can create according to that particular meta-model
  - A meta-model allows to talk about semantics

- Meta-modeling allows one to customize modeling formalisms to the habits of individual modelers and users to specify the relationships among models expressed in different modeling formalisms or domain ontologies.

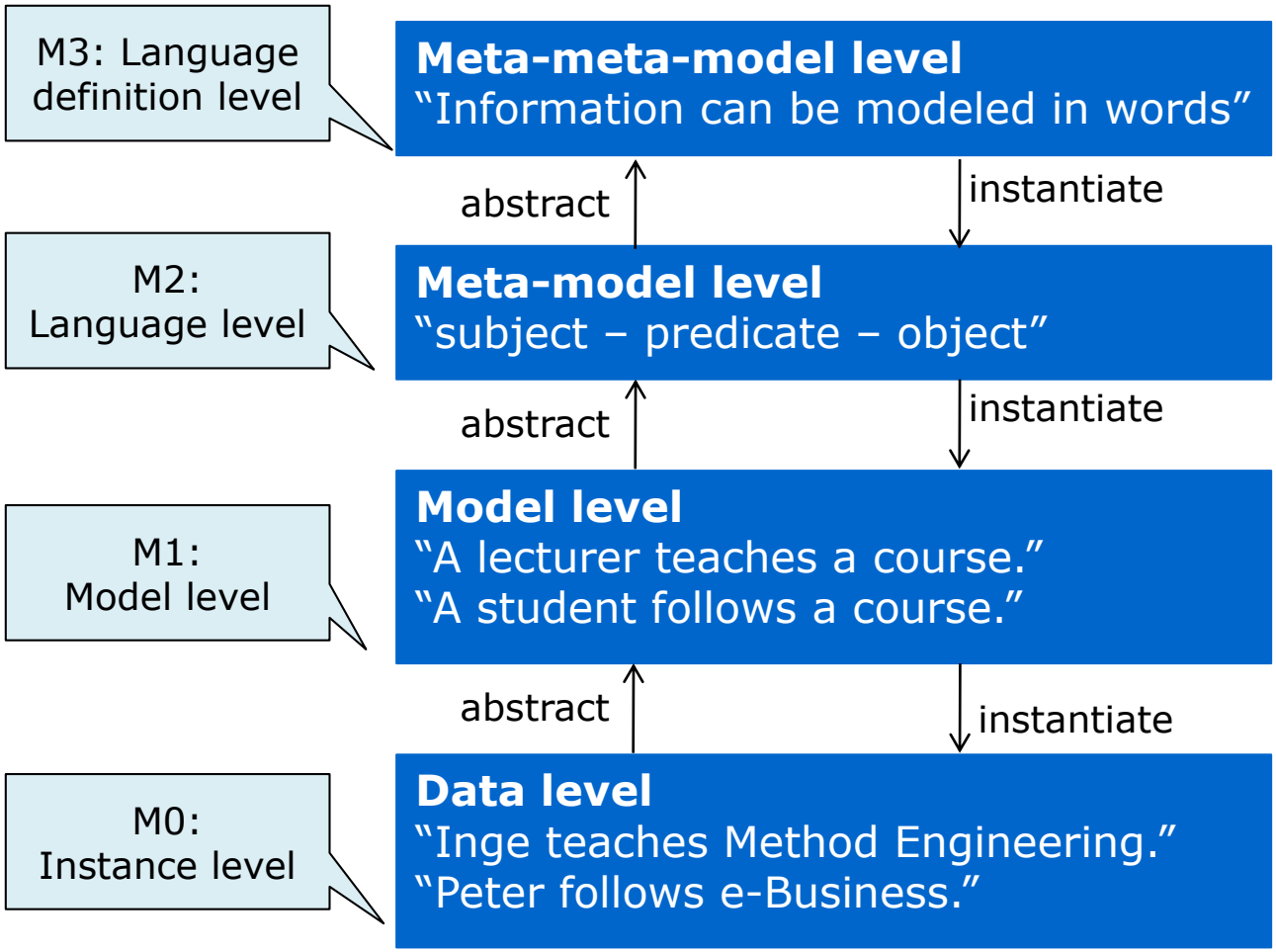
IRDS meta-modeling framework

- Meta-modeling is not new
- International Organization for Standardization (ISO) designed the Information Resources Dictionary System (IRDS) Standard (ISO/IEC 1990)



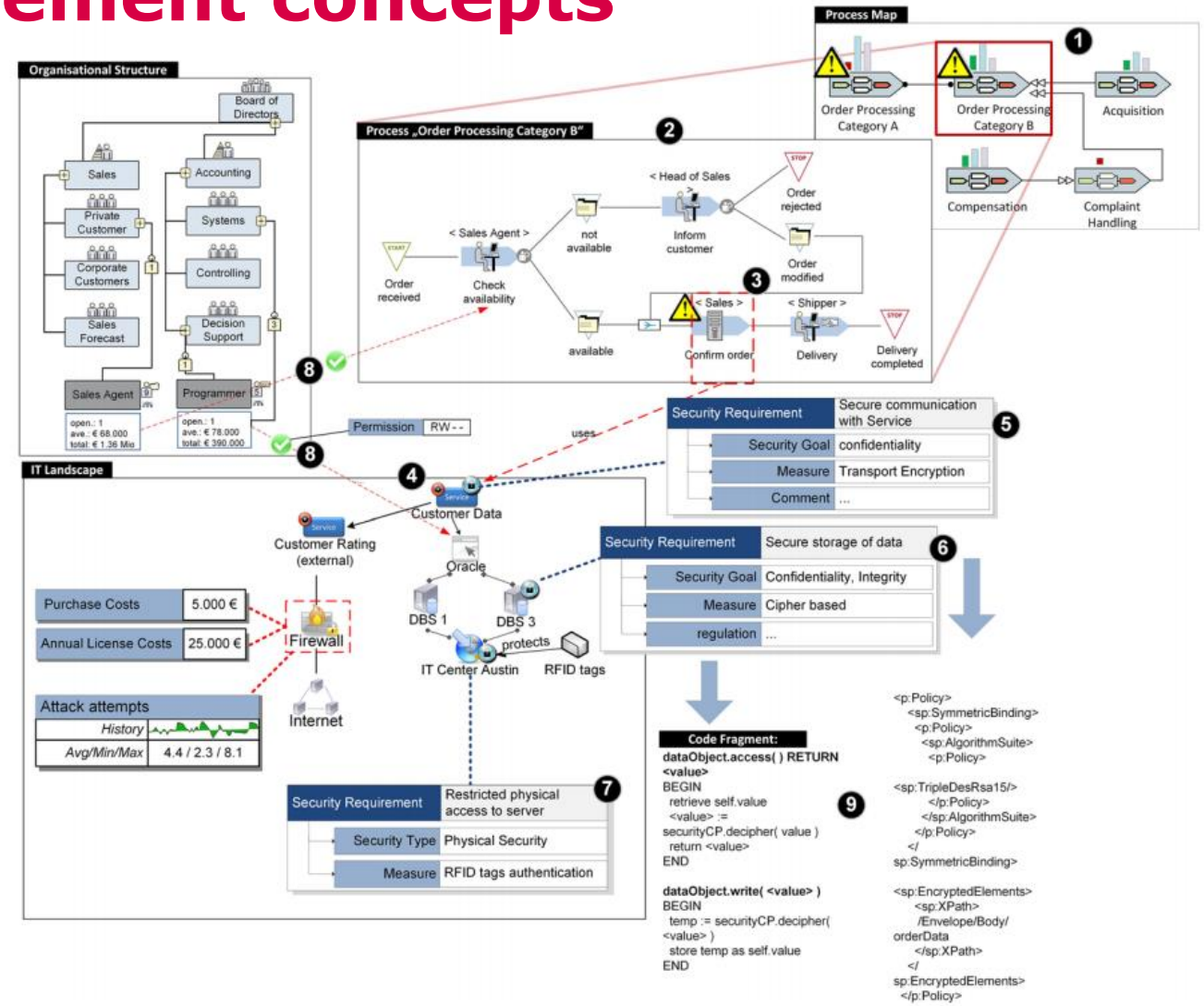
Goldfine and Konig (1985)

Abstractions in statements about artifacts



Jeusfeld (2010)

Enterprise models enriched with security management concepts



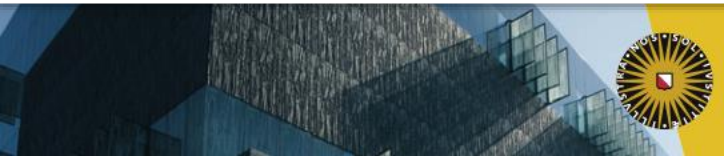
Goldstein and Frank (2016)

UML/MOF example

- 1997: Object Media Group (OMG) proposed UML 1.0
  - Appeared to have a lot of semantic problems
- The OMG tried to deal with these problems by formalizing the language
  - Idea: Use meta-modeling!
- The OMG realized that all that was needed to describe meta-models was to use a subset of UML class diagram elements
  - Insight: To describe any meta model, the UML class diagram notation itself can be used

Meta-Object Facility (MOF)

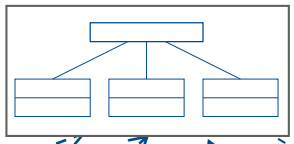
- In UML-2, the OMG introduced the MOF to create a common approach to meta-modeling
- A meta-model which is defined using MOF is called MOF compliant.
- Advantages of using MOF-compliant models:
  - They can easily be compared
  - MOF compliant models can be exchanged in a standardized way (XML Metadata Interchange)
  - MOF compliant instances can live in the same metadata repository (data warehousing)



OMG (2006)

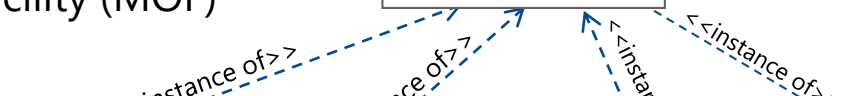
Theoretical foundation: The UML language architecture

Level M3: Meta Meta Model
OMG Meta-Object Facility (MOF)

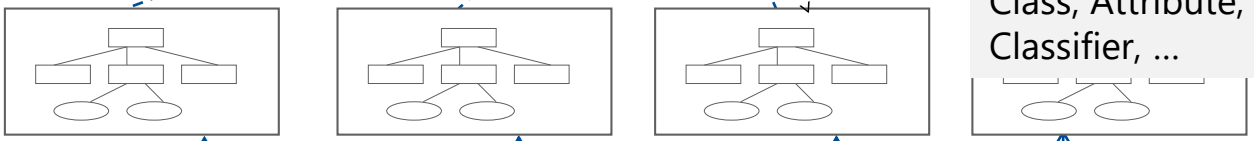


Class, Property, Association, …

‚Metaclass'



Level M2:
Meta Model
OMG UML



Class, Attribute, Classifier, …

UML Class Diagram

<<instance of>>

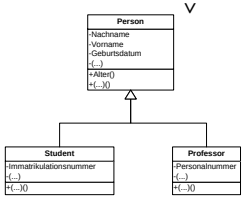
UML Use Case Diagram

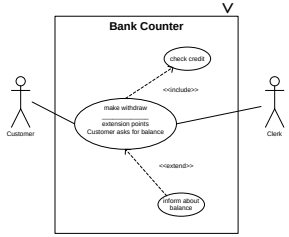
<<instance of>>

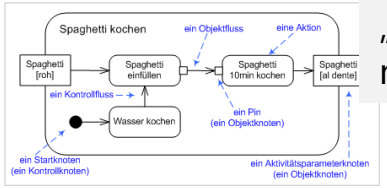
UML Activity Diagram

<<instance of>>

Level M1:
Type Models







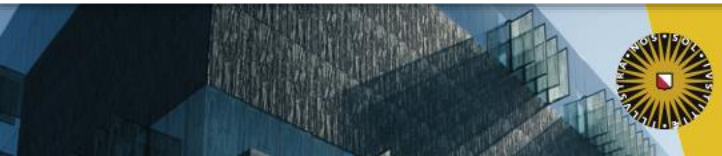
„Customer", „First name",

Level M0:
Instances

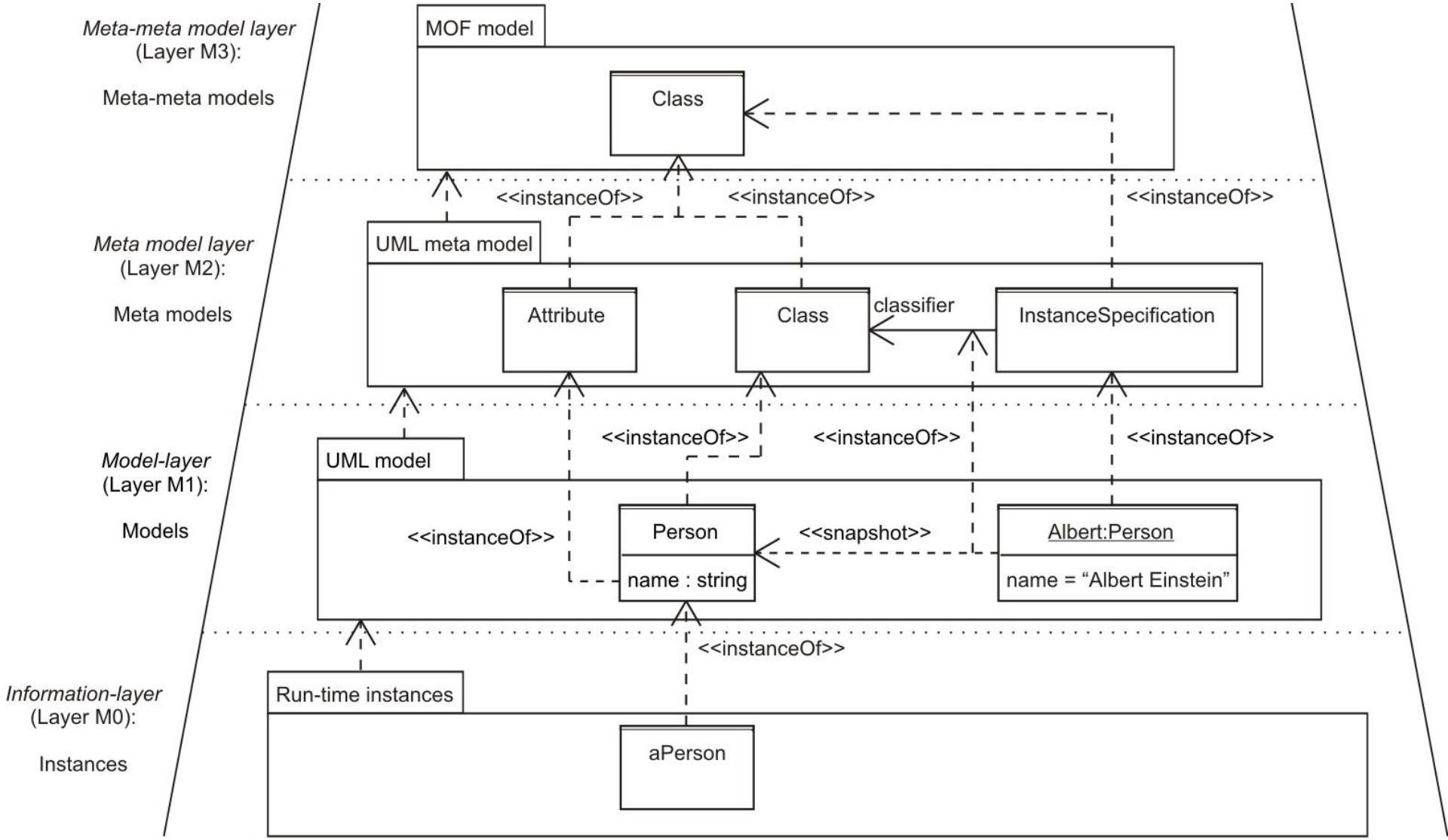
„ABZ AG", "Order No. 4711 of 04/20/2011, 10:30am", „43123"

MOF hierarchy

- We have seen they have used the IRDS modeling sequence data → model → meta model → meta-meta model
- This sequence could be continued infinitely, but four models are enough for most modeling purposes.
- MOF defines a four-layer meta model hierarchy
  - Layer M3: Meta-meta model layer (The MOF model)
  - Layer M2: Meta model layer (E.g., the UML CD meta model)
  - Layer M1: Model layer (A UML model, such as a class diagram)
  - Layer M0: Information layer (the Application)

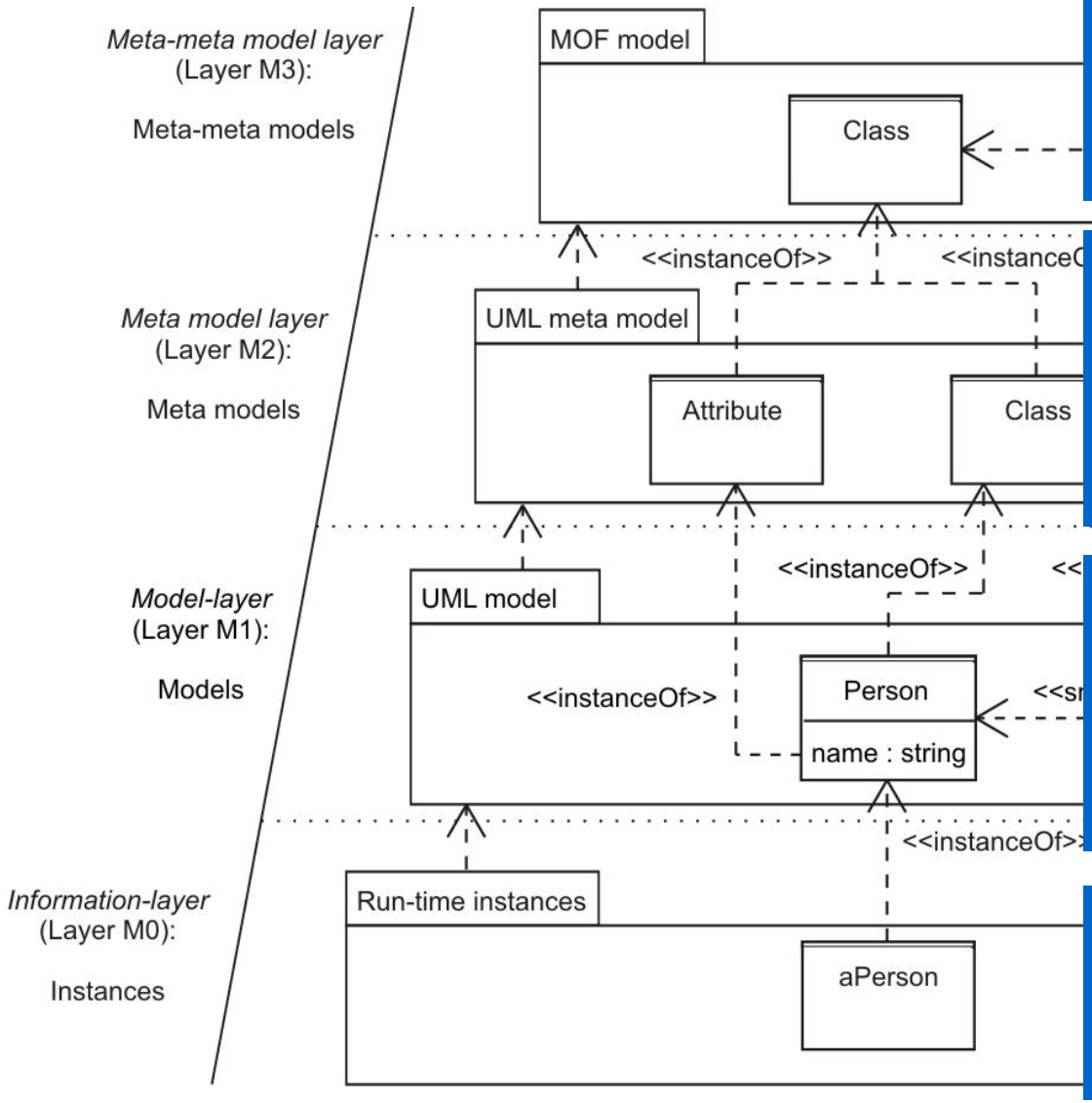


OMG (2006)



MOF hierarchy for UML class diagrams

Bruegge and Dutoit (2003)



Layer M3: MOF model

The UML meta model is an instance of a yet more abstract model, namely the *MOF model*

➔ Layer M3 contains *meta-meta models*

Layer M2: UML meta model Models the language UML, i.e., defines concepts like *classes*, *attributes*, *associations*

➔ Layer M2 contains descriptions of elements that can be used to describe the models on the model layer

Layer M1: UML-model
By using any UML diagram, we instantiate the UML meta model and obtain a UML model

➔ Layer M1 contains *application-specific* models

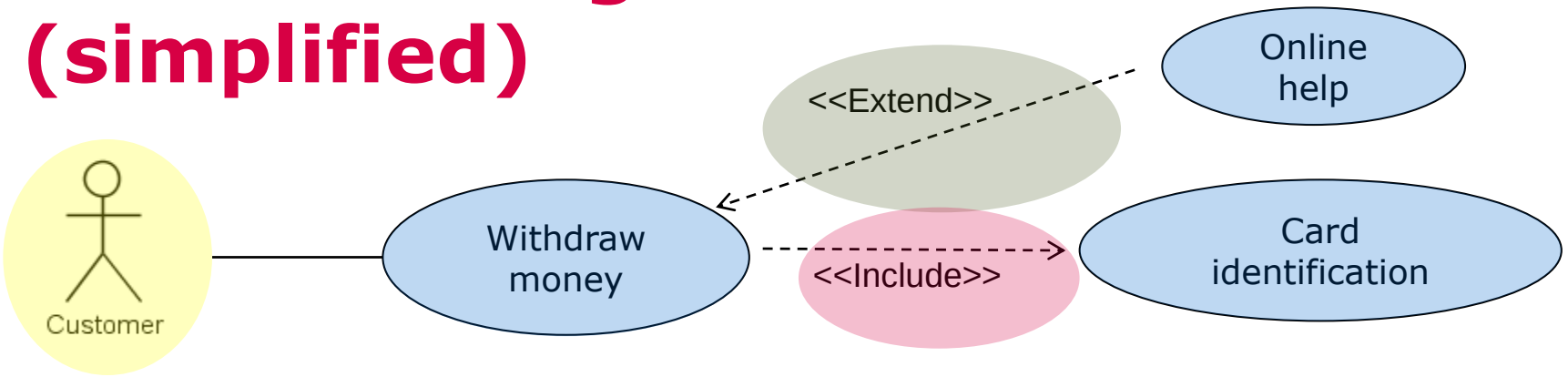
Layer M0: Run-time Instances Real instances of the models

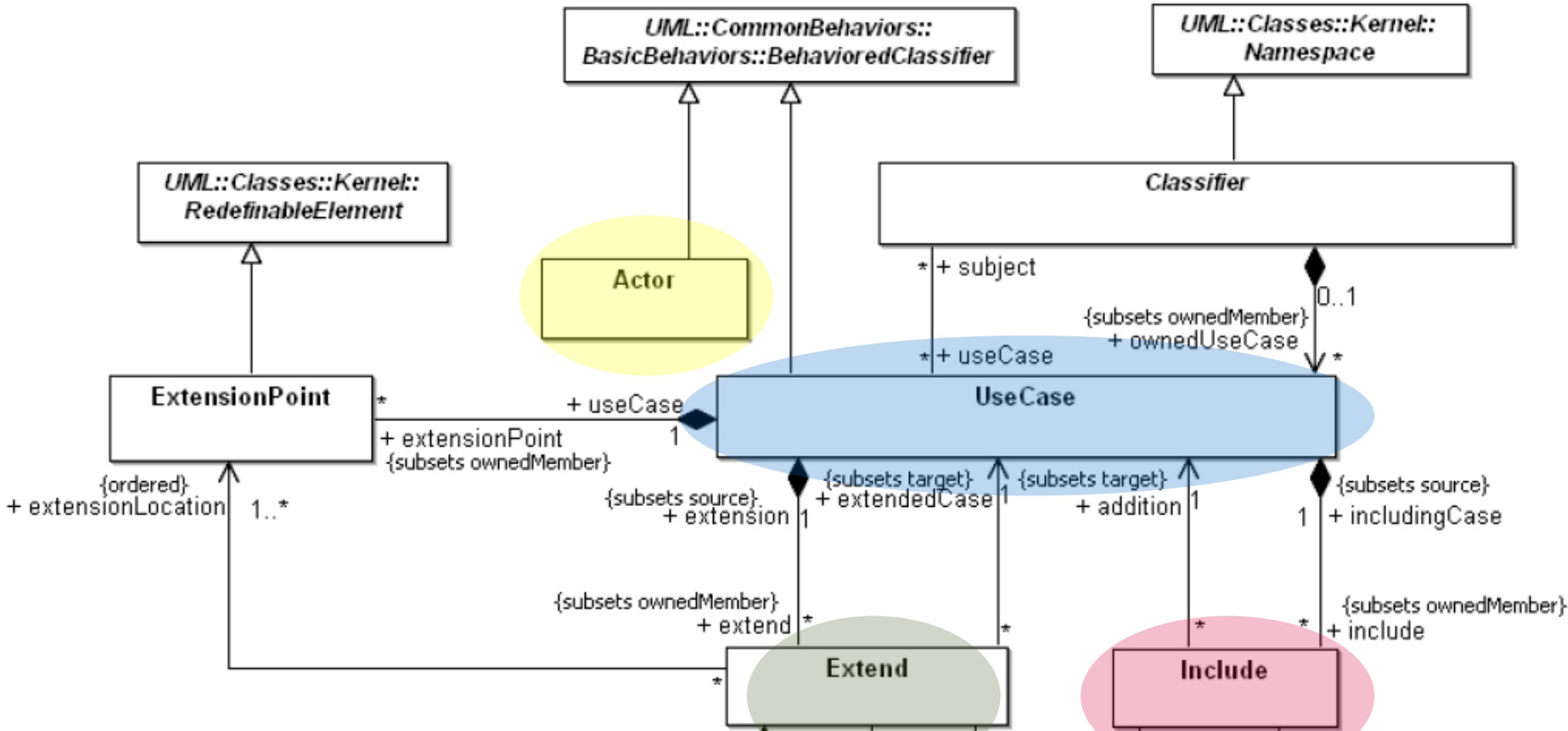
➔ *Layer M0 contains concrete run-time instances*

MOF hierarchy for UML class diagrams

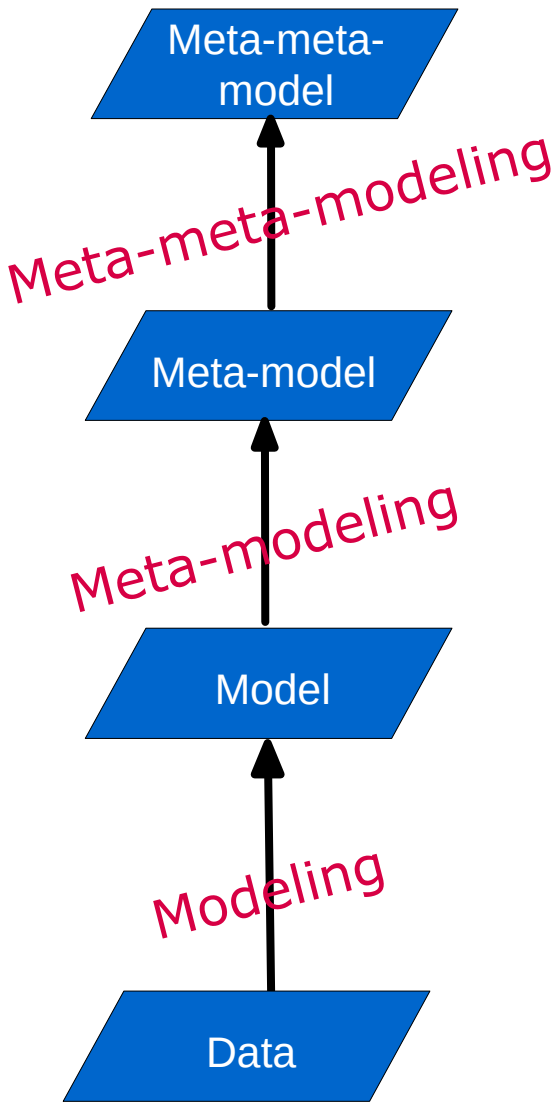
Bruegge and Dutoit (2003)

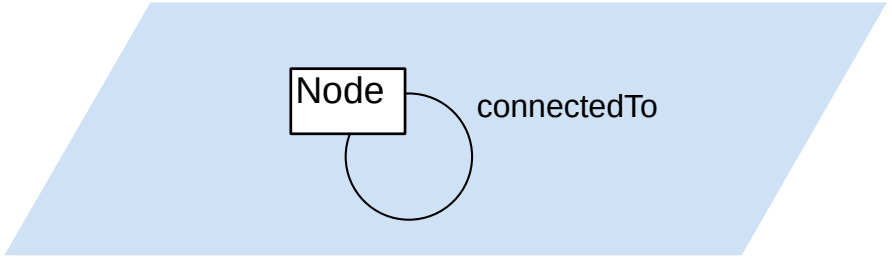
Use Case Diagram meta-model (simplified)



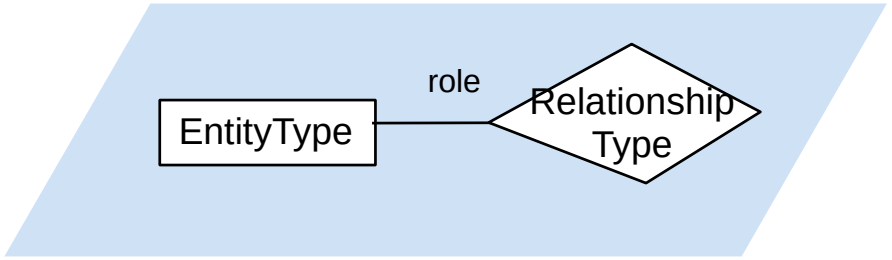


ERD-example

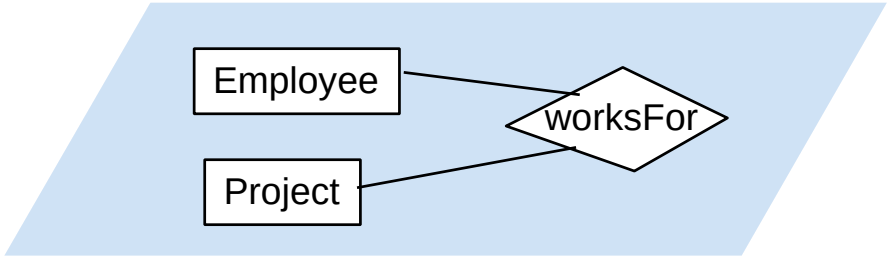




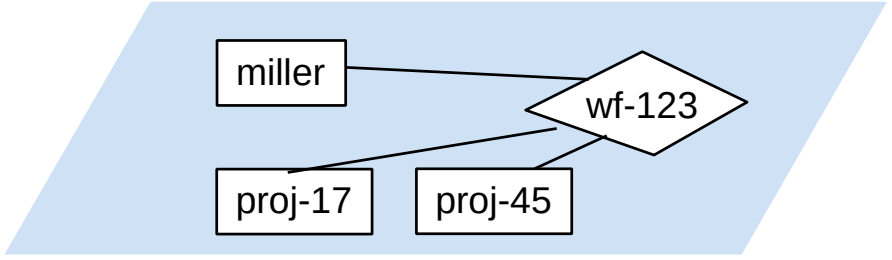
... defines node and link types



... defines entity types etc. by distinct node and link types



... defines the conceptual schema of a database in ER notation



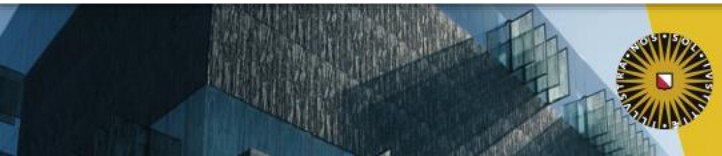
... contains example data conforming the schema

Jeusfeld (2010)

Meta-data modeling

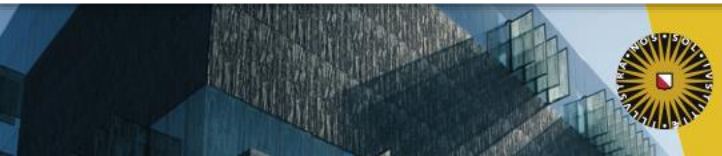
- A meta-data model exposes the concepts and rules of a technique or method

- Usage:
  - Reasoning about the properties of a method or technique
  - Comparison of methods or techniques
  - Development of systems development tools
  - Relationships and integrations of techniques
  - Assembly of situational methods



Techniques for meta-data modeling

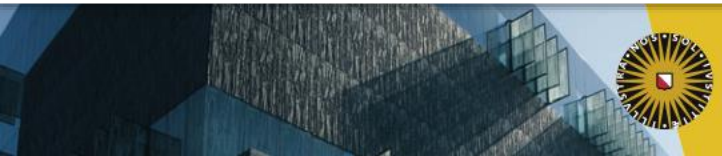
- Several data modeling techniques are in use for meta-modeling:
  - Entity-Relationship diagrams (ERD)
  - UML class diagrams
  - Object-Property-Role-Relationship (OPRR) diagrams
  - …



Meta-data modeling with process-deliverable diagrams

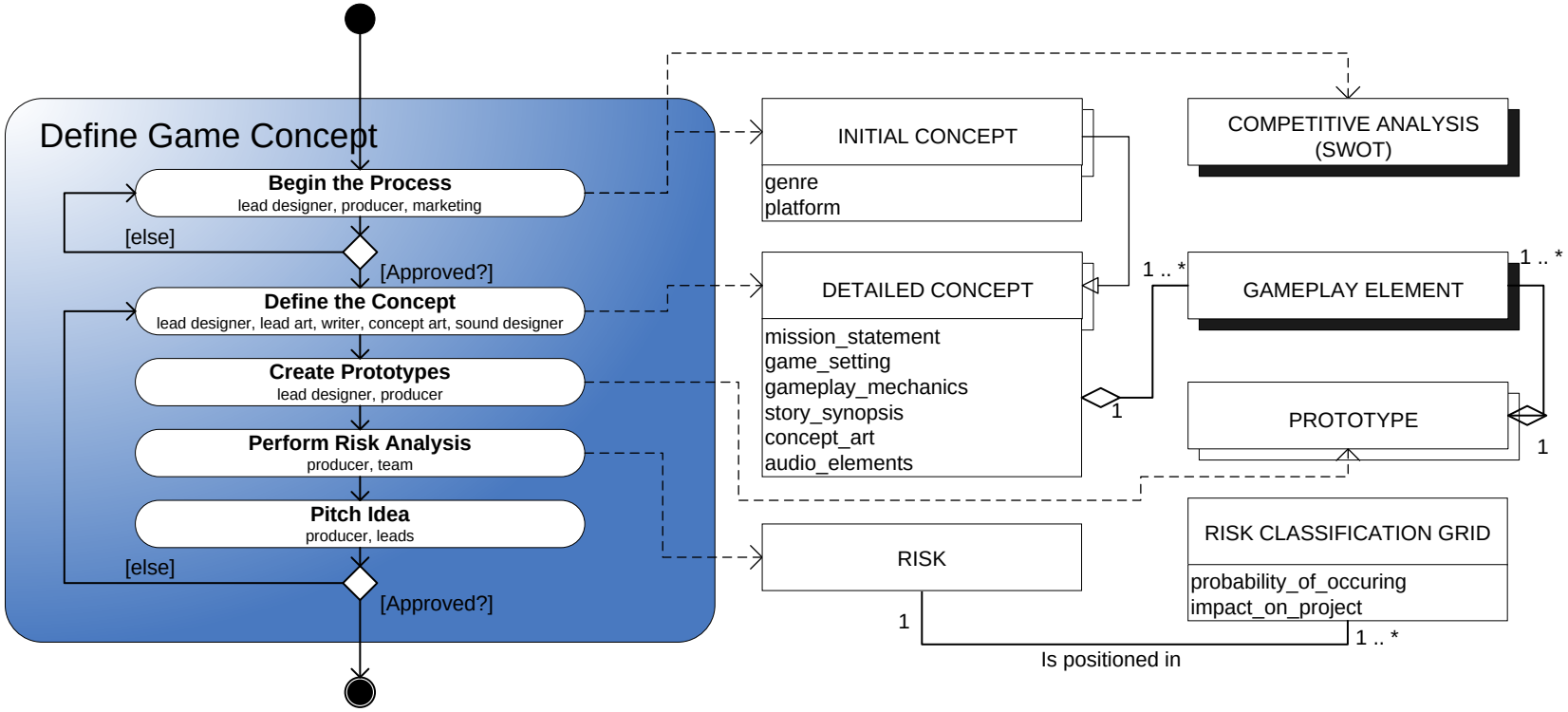
- In the method engineering course, we use process deliverable diagrams (PDDs)

- PDDs are based on UML activity diagrams and UML class diagrams
  - However, we use a simplified version
  - PDDs were formerly called process data diagrams



van de Weerd and Brinkkemper (2008)

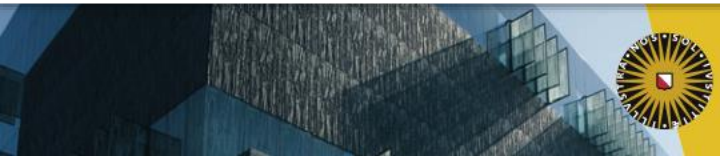
Example – Process-deliverable diagram



- Process-deliverable diagram of the Concept Phase in "The Game Production Handbook"

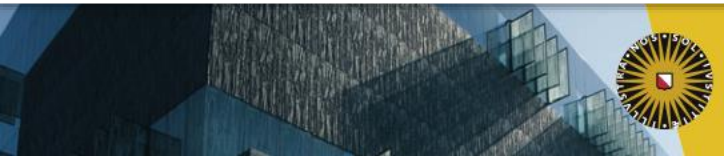
Agenda

- Introduction to meta-data modeling
- PDD notation
- Examples and assignments



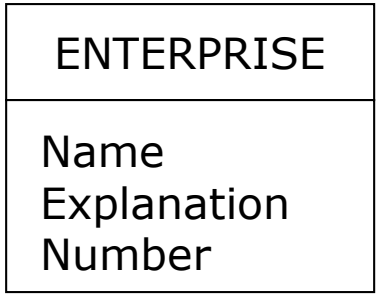
Notation

- Concepts
- Complex concepts
- Associations
  - Multiplicity
  - Non-binary associations
  - Recursive associations
  - Ternary associations
- Aggregation
- Generalization
- Version dependent data



Concept

Concept
(NAME in CAPITALS and
always in singular form)



Concept with properties
(keys may be underlined if relevant)

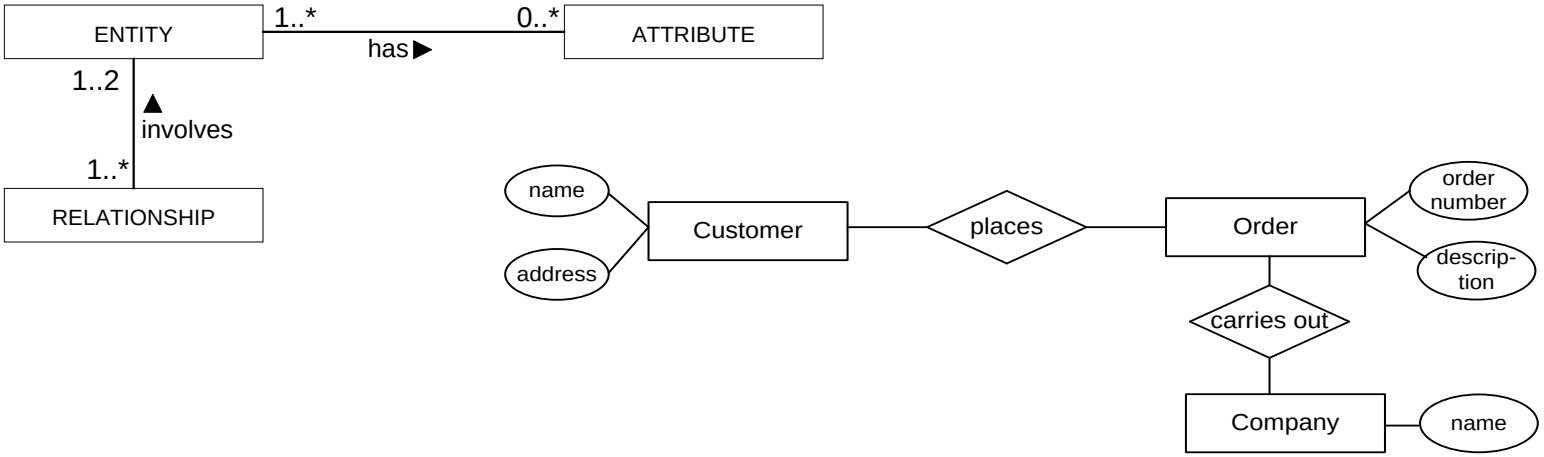
Definition

A concept is defined as an independent notion in a method or technique that is *instantiated* while executing a method or technique.

Concept examples

- Examples: ENTITY, ENTERPRISE, DATA FLOW

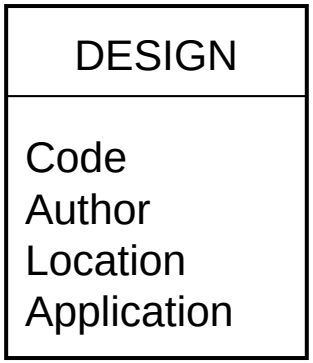
- E.g. during data modeling using the Entity Relationship Diagramming technique the concept ENTITY is instantiated with “Customer”, “Order” and “Article”.



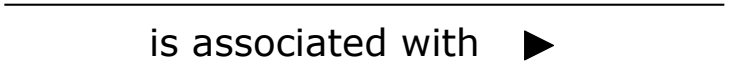
Concept properties

- Sometimes the need exists to assign properties to concepts.
- Properties are written in lower case, under the concept name.

- Example



Association



Definition

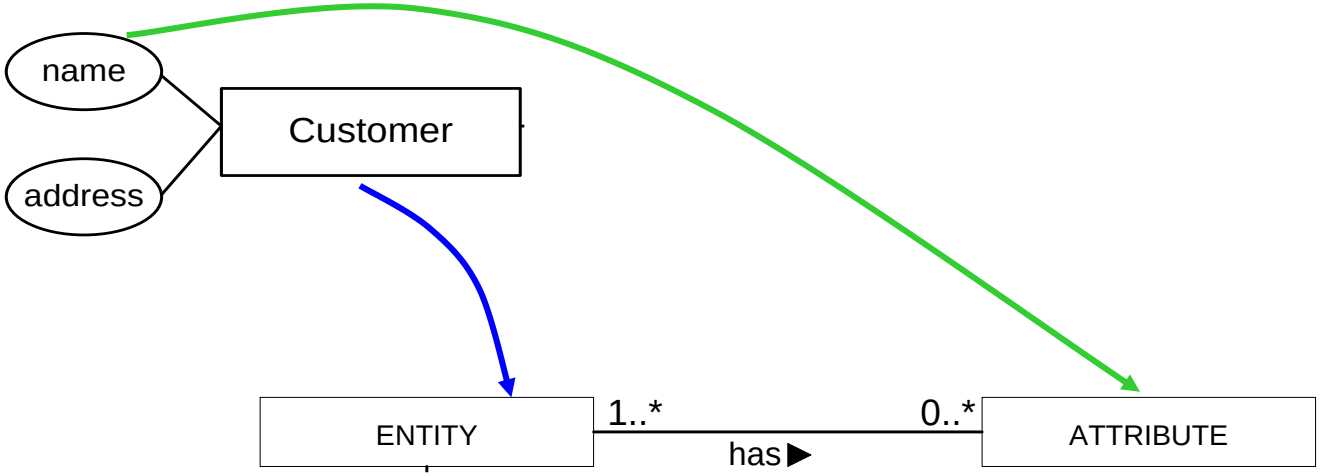
An association is defined as a relationship between two or more concepts, or one concept and itself.

During modeling, instances of concepts are *first* identified and *then* related

Association example 1

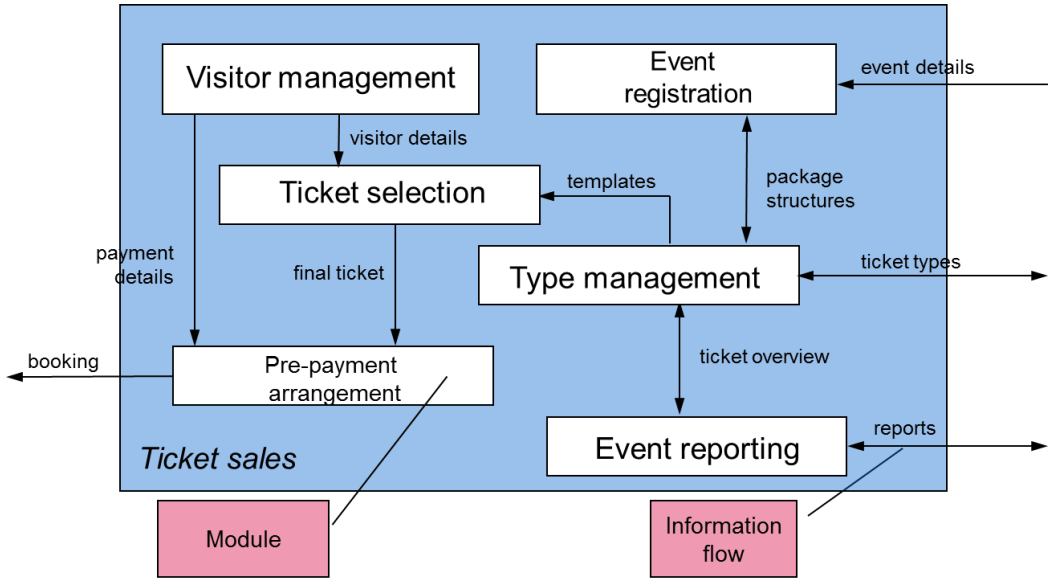
Example entity-relationship diagram

- ENTITY “Customer” has ATTRIBUTEs “name” and “address



Association example 2

Example Functional Architecture diagram



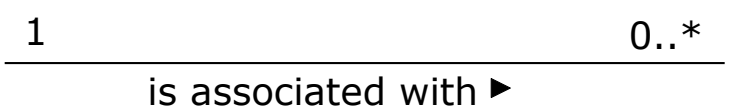
MODULE is destination of INFORMATION FLOW

Multiplicity

Purpose

To indicate the multiplicity of instances of a certain concept

To be used on either end of an association:



1 for exactly one
0..1 for one or zero
0..* for zero or more
1..* for one or more

Multiplicity example 1 (M1)

Model the following

A professor teaches 0 or more courses. Each course is taught by at least 1 professor. (M1-level)

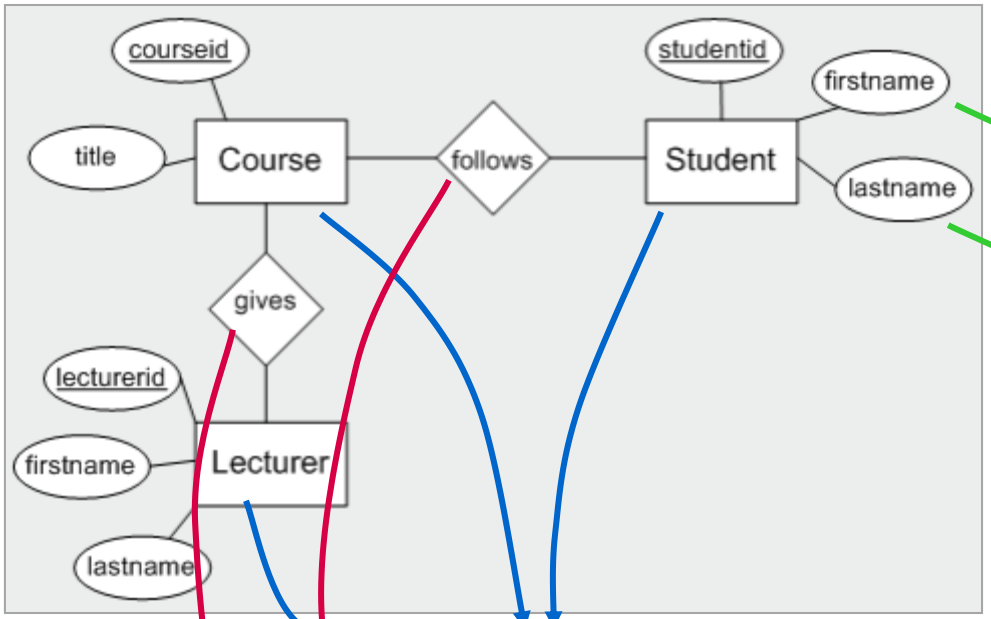
Multiplicity example 2 (M2)

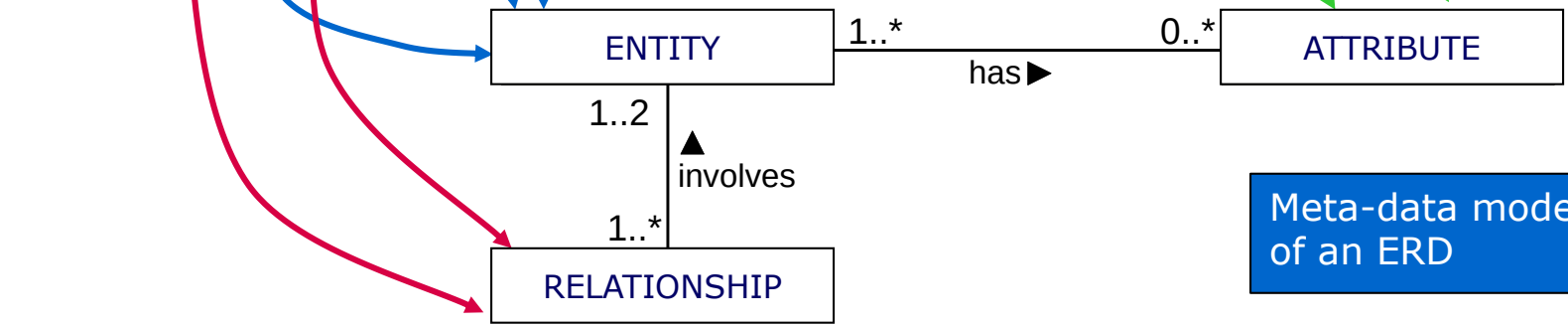
Model the following

An ENTITY is described by at least 1 ATTRIBUTE.
An ATTRIBUTE describes exactly 1 ENTITY.

Creation of a meta-data model

Binary Entity Relationship diagram



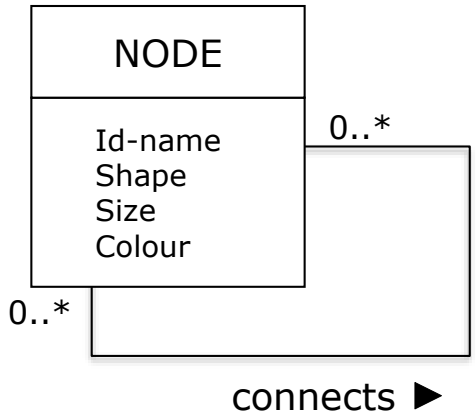


Meta-data model of an ERD

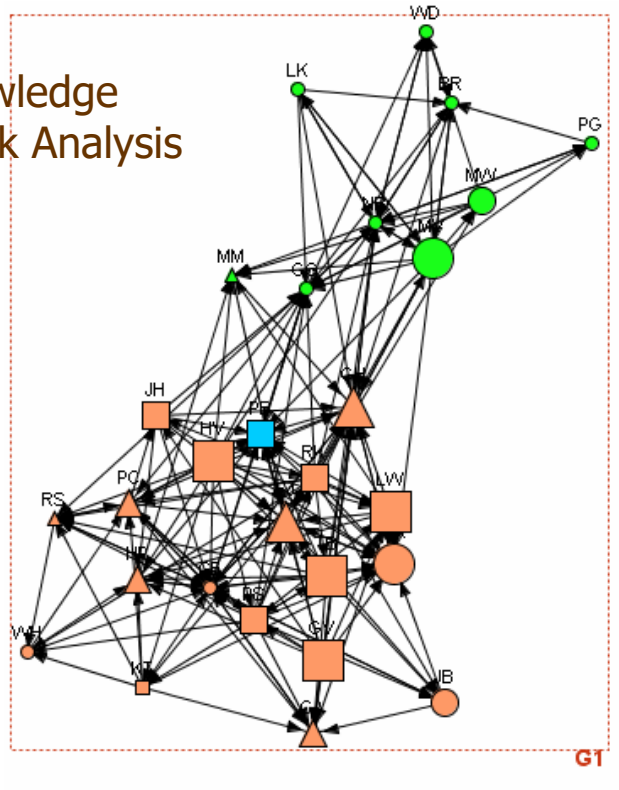
Non-binary associations (1)

Model the following:

Nodes can be connected to other nodes



Knowledge Network Analysis



Recursive association

Non-binary associations (2)

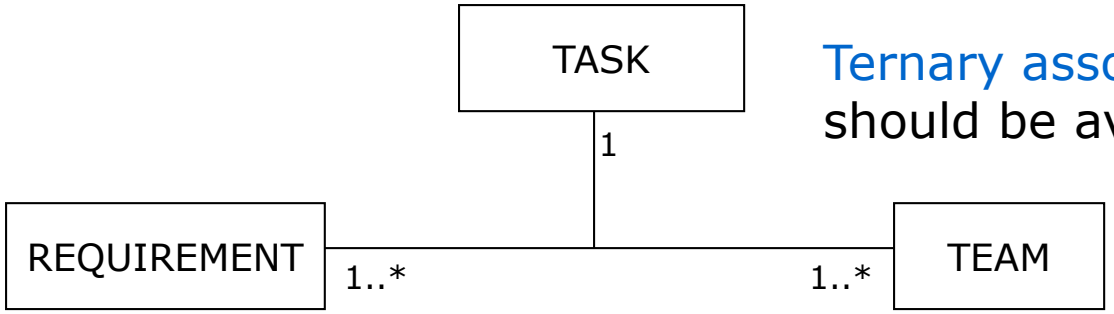
Model the following:

An assignment of a project plan consists of a team (to which the task is assigned), a requirement (which is being worked upon in the task), and a task (work on the requirement).

Project plan Assignments

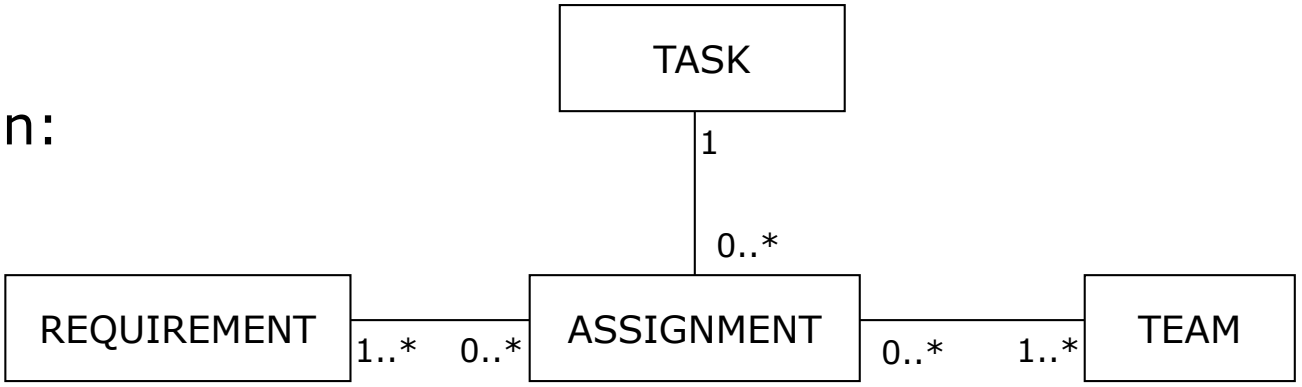
Requirement	Team A	Team B	Team C
Autorisation on order cancellation and removal	5		45
Autorisation on archiving service orders	2	5	5
Performance improvements	15		
Inclusion grafical plan board	10	10	50
Link with Acrobat reader for PDF files		33	
Optimalising interface with international Postal code system			15
Adaptations in rental and systems		20	20
Adaptations in symbol import	10		
Comparison of services per department		9	25
Totals	**42**	**77**	**160**

Non-binary associations (3)



Ternary association (note: rare, should be avoided)

Preferred option:



Why?

- Cardinality and participation of association unclear

Aggregation

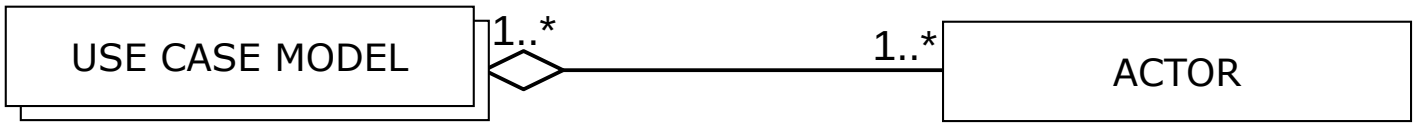
Definition

Aggregation represents the relation between a concept (as a whole) containing other concepts (as parts).

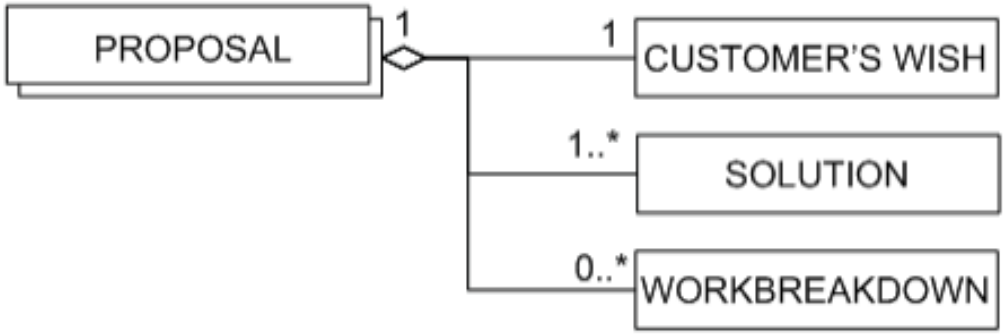
Aggregation is a special type of association.

Aggregation examples

- USE CASE MODEL consists of ACTORs



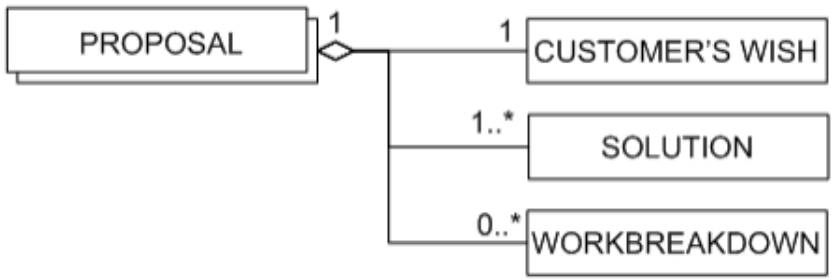
- PROPOSAL consists of a CUSTOMER'S WISH, 1 or more SOLUTIONs and 0 or more WORK BREAKDOWN



Concept or property?

- Ask yourself the following questions:
  - Is the relationship between the two elements a 1-1 relationship?
  - Is the concerning element an 'atomic' element?
- Yes to both questions? → model as property
  - Especially for elements such as *version number, id, author, label*, etc.
- Should it be decomposed into other elements? Then a concept

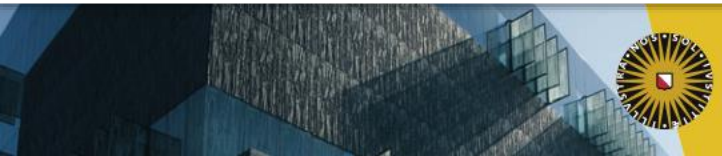
- However, distinction is not always crystal clear:



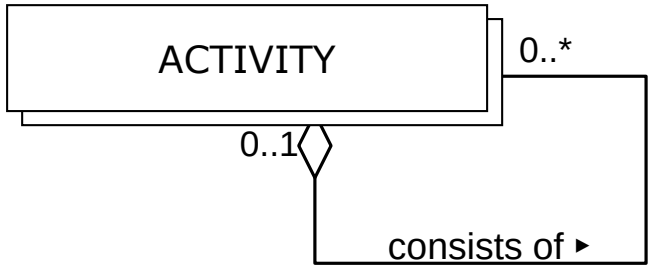
Association or aggregation?

- The decision to use association or aggregation is a matter of judgement and is sometimes arbitrary.

- Example:
  What type of relationship should be used to model a car with its tires?
  - If the application is a service center → aggregation
  - If the application is a tire store → association

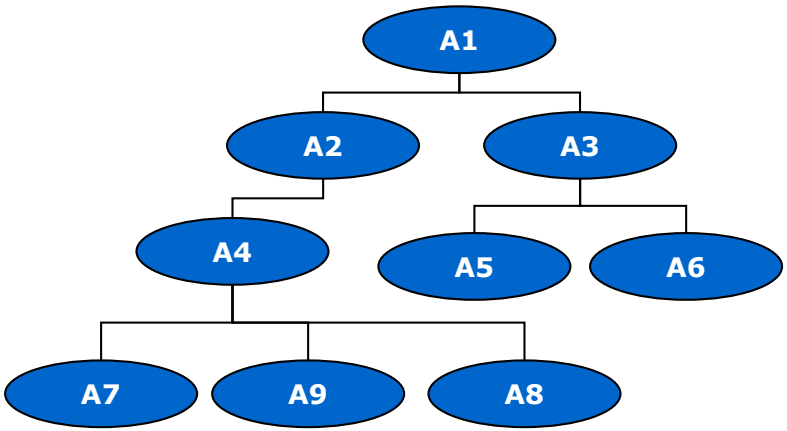


Recursive aggregation



- An ACTIVITY belongs to 0 or 1 ACTIVITIES
- An ACTIVITY can have 0 or more sub ACTIVITIES.

Activity tree



Questions:

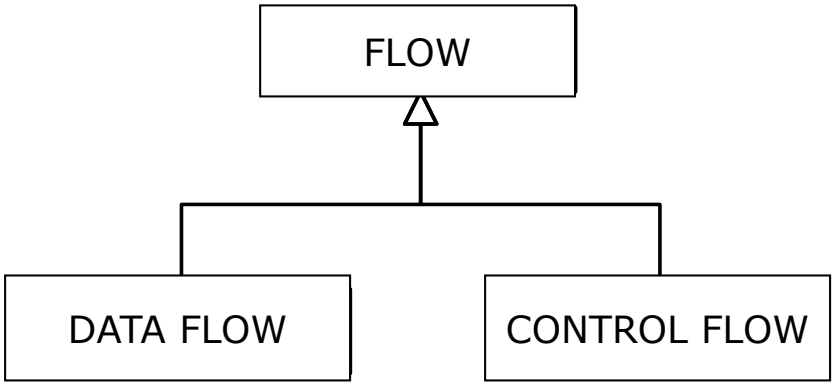
- What would you change in the meta-model to make a connection possible between A5 and A9?
- What would happen if you changed both zeros in the multiplicities of the meta-model to 1?

Generalization

Purpose:

- To express a relationship between a general concept and a more specific concept.

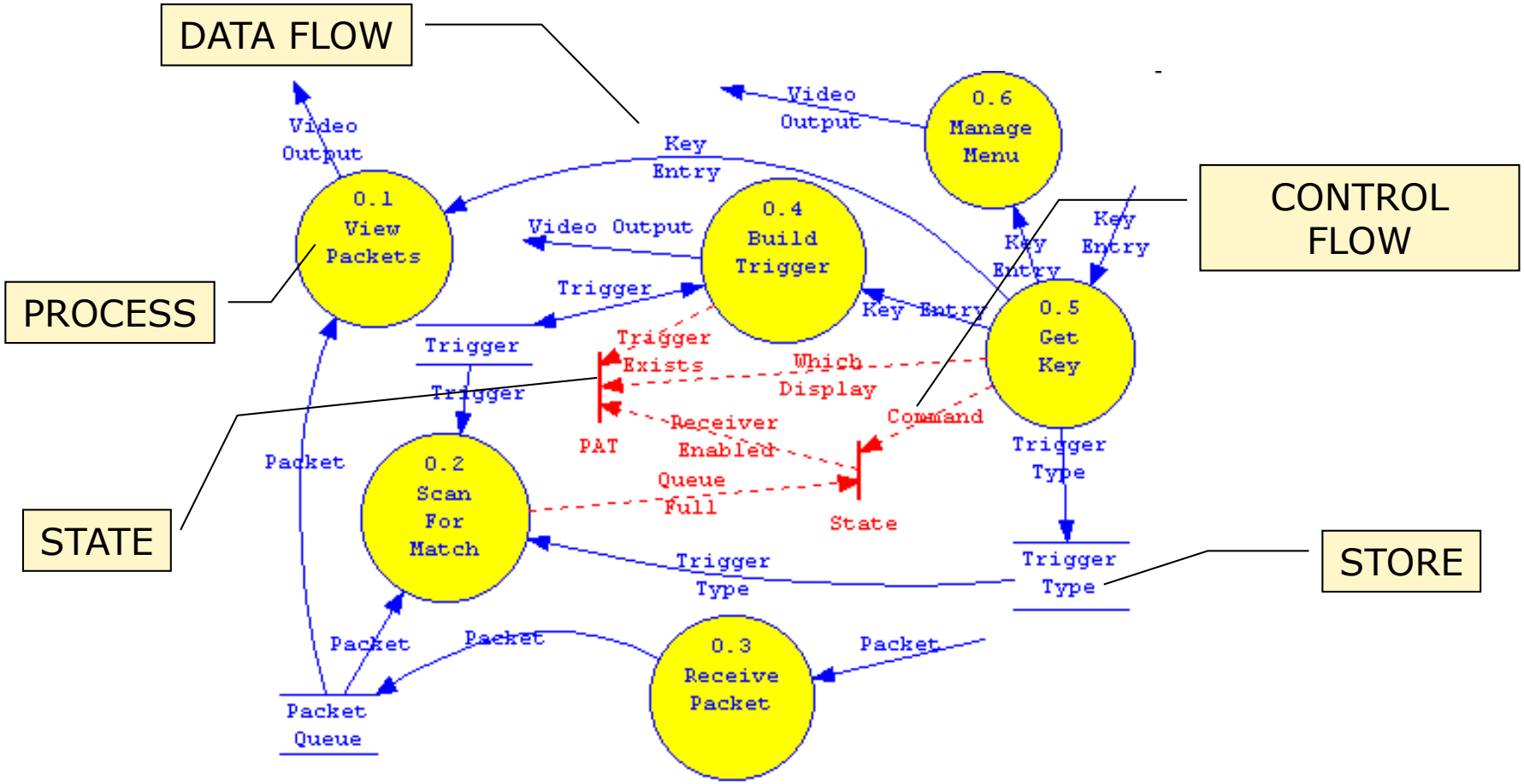
- The more specific concept is fully consistent with the more general concept and may contain additional information.



Example:

a DATA FLOW is a FLOW

Generalization example

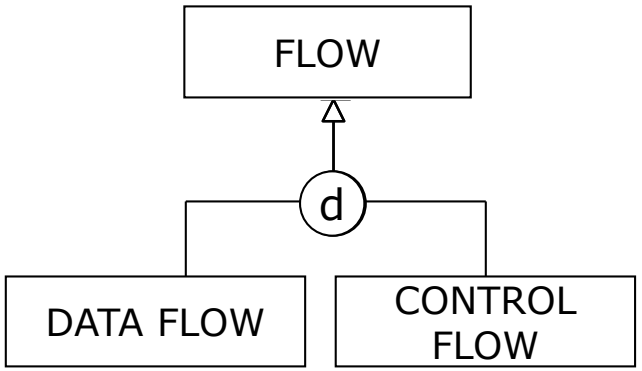


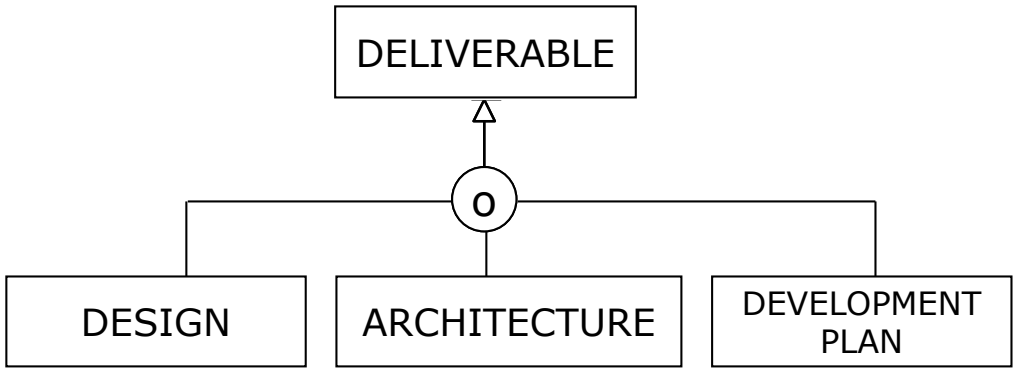
Data Flow Diagram

Additional rules for generalization

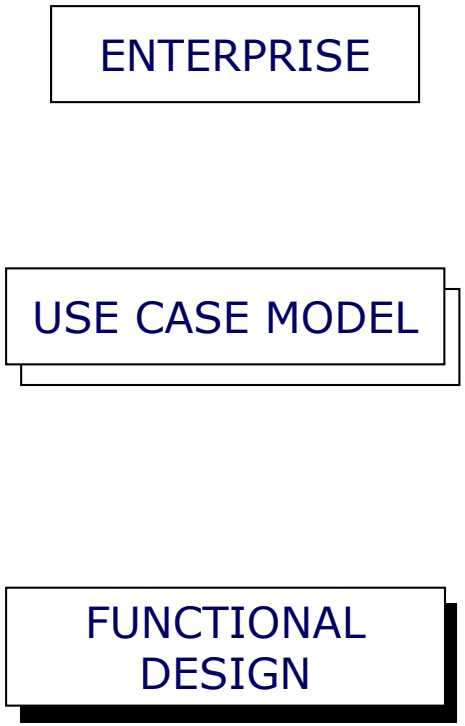
- d = disjoint: occurrence of one concept is incompatible with the occurrence of the other concept
- o = overlapping: occurrence of the concepts may overlap with the occurences of the other concept(s)
- c = categories: the disjoint concepts have no occurrences in common and that the decomposition is exhaustive

Examples:





Types of concepts

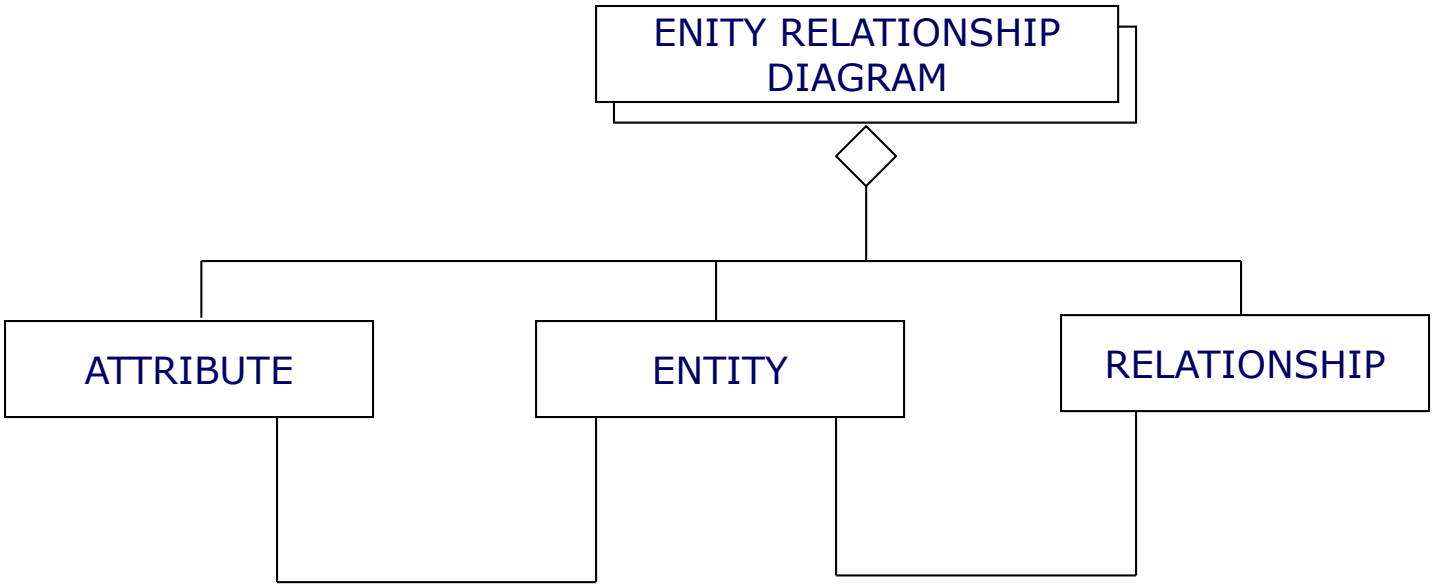


Standard Concept without sub-concepts (atomic concept)

Complex concept with sub-concepts

- Open concept: subconcepts are *known* in this context: i.e. the meta-model details can be found in this or another meta-model of the method engineering study at hand.
- Closed concept: subconcepts are *not known or not relevant* in this context

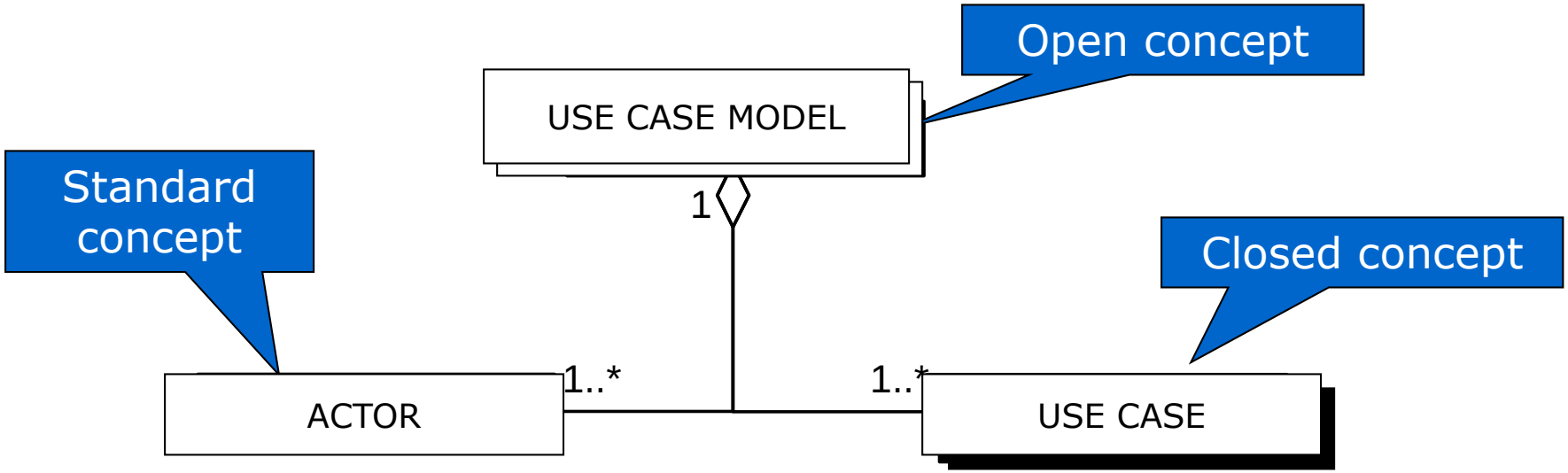
Complex concepts



- A complex concept is an aggregate of concepts and possible associations for representing complete or partial products (so-called product fragments) in a method

Aggregation and complex concepts

Aggregation always requires the use of an open concept.



Open and closed concepts serve as a means for hiding details in large meta-models

Wrap-up

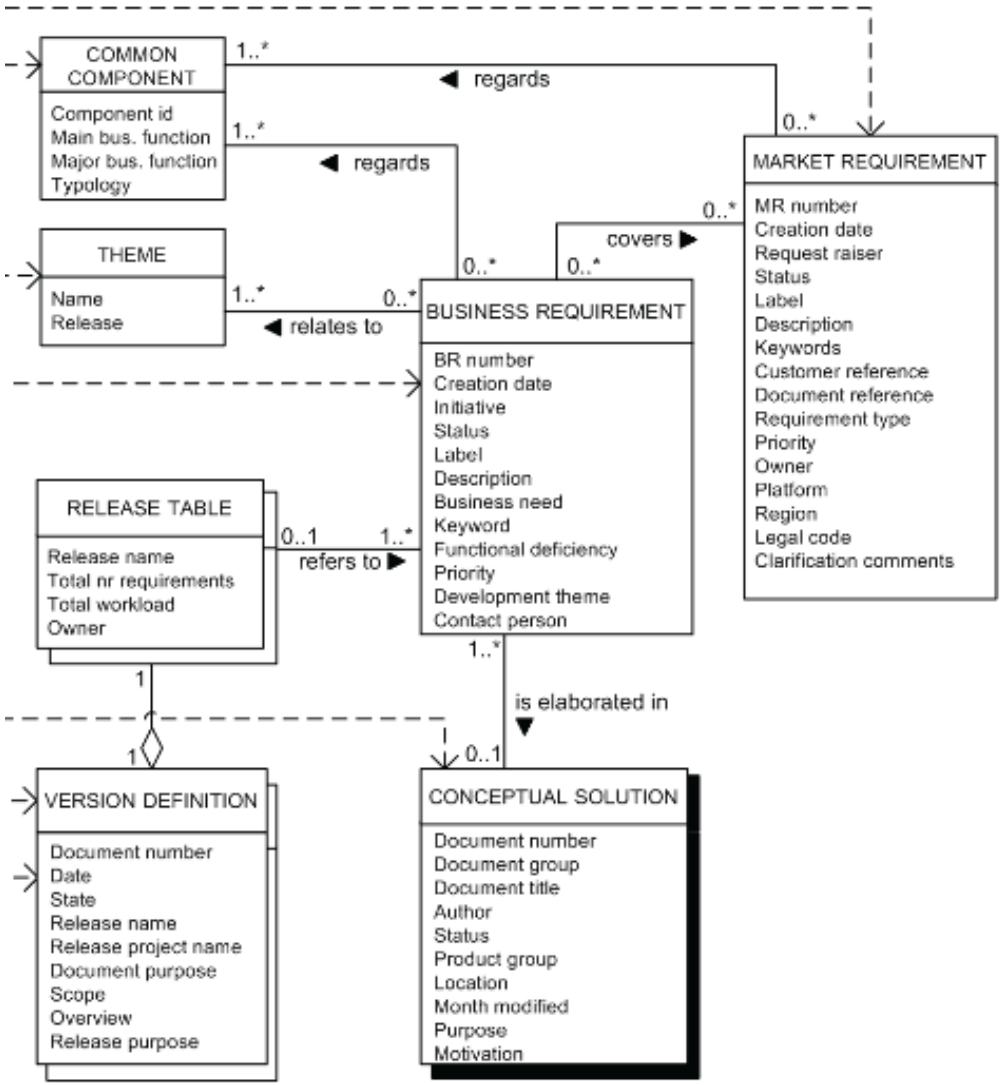
- Concepts
- Associations
  - Multiplicity
  - Non-binary associations
  - Recursive associations
  - Ternary associations
- Aggregation
- Generalization
- Complex concepts

Agenda

- Introduction to meta-data modeling
- PDD notation
- Examples and assignments

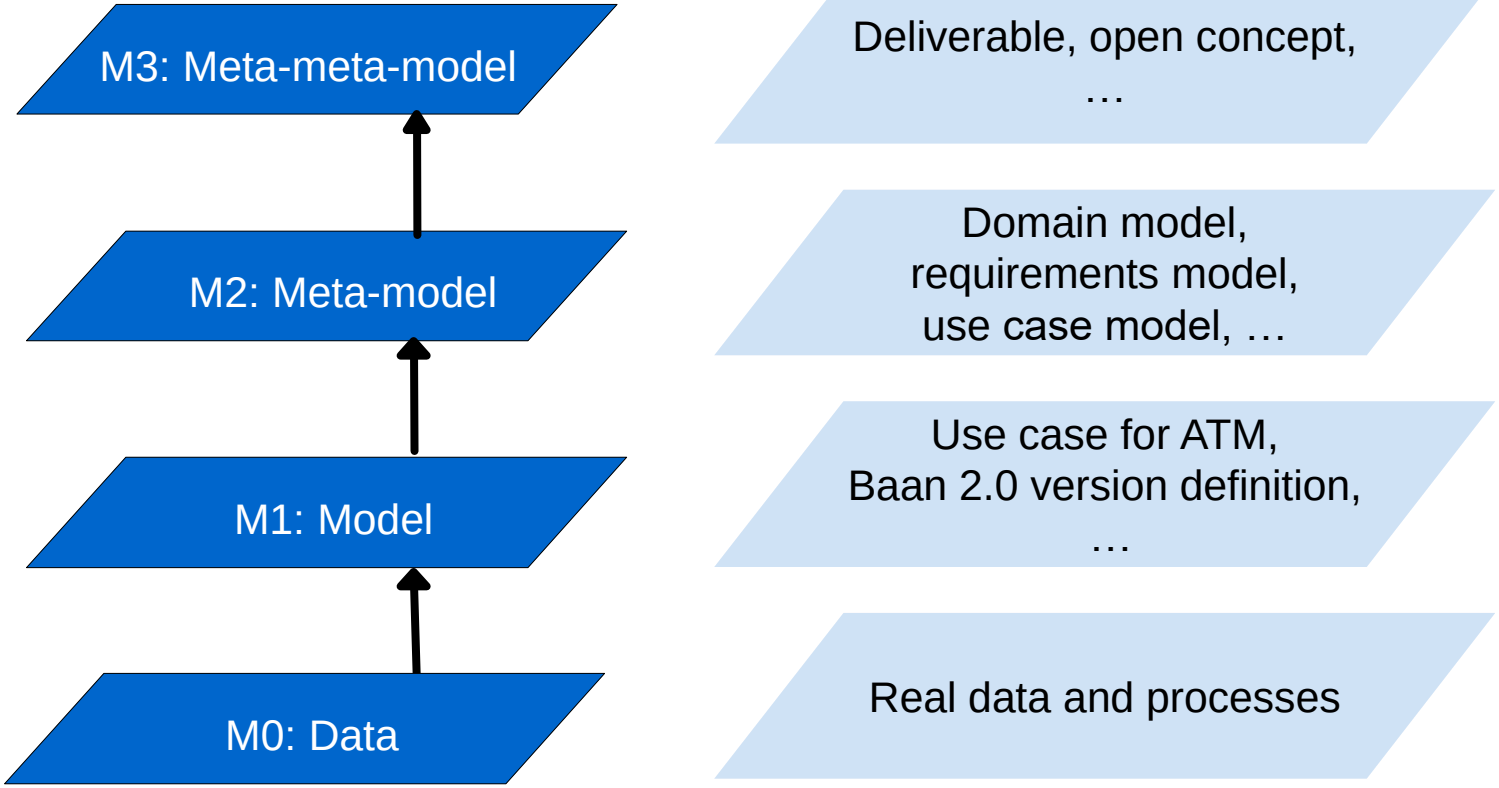
Example

PDD of the requirements and release process at Baan
(Van de Weerd et al., 2010)

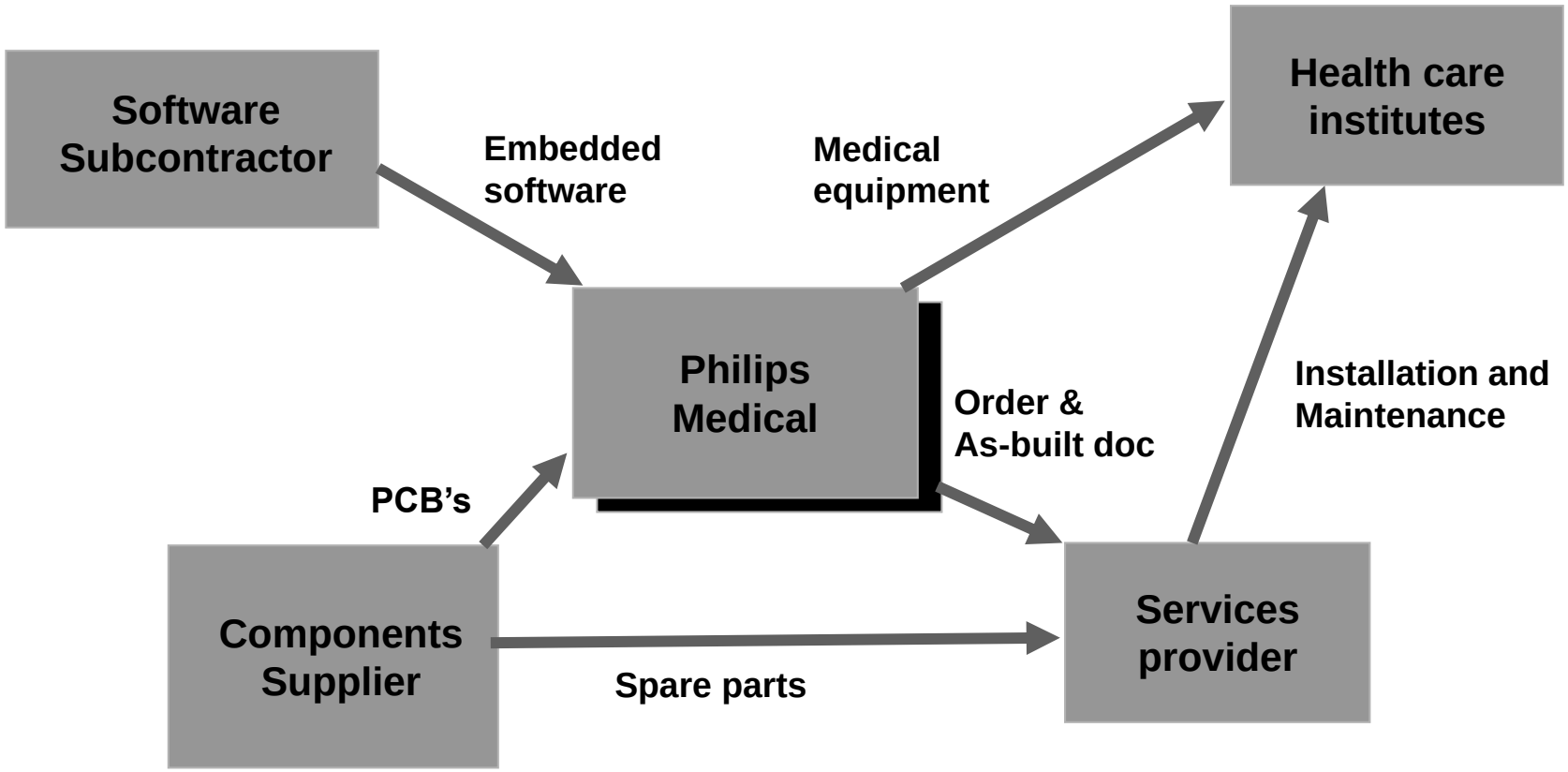


Question

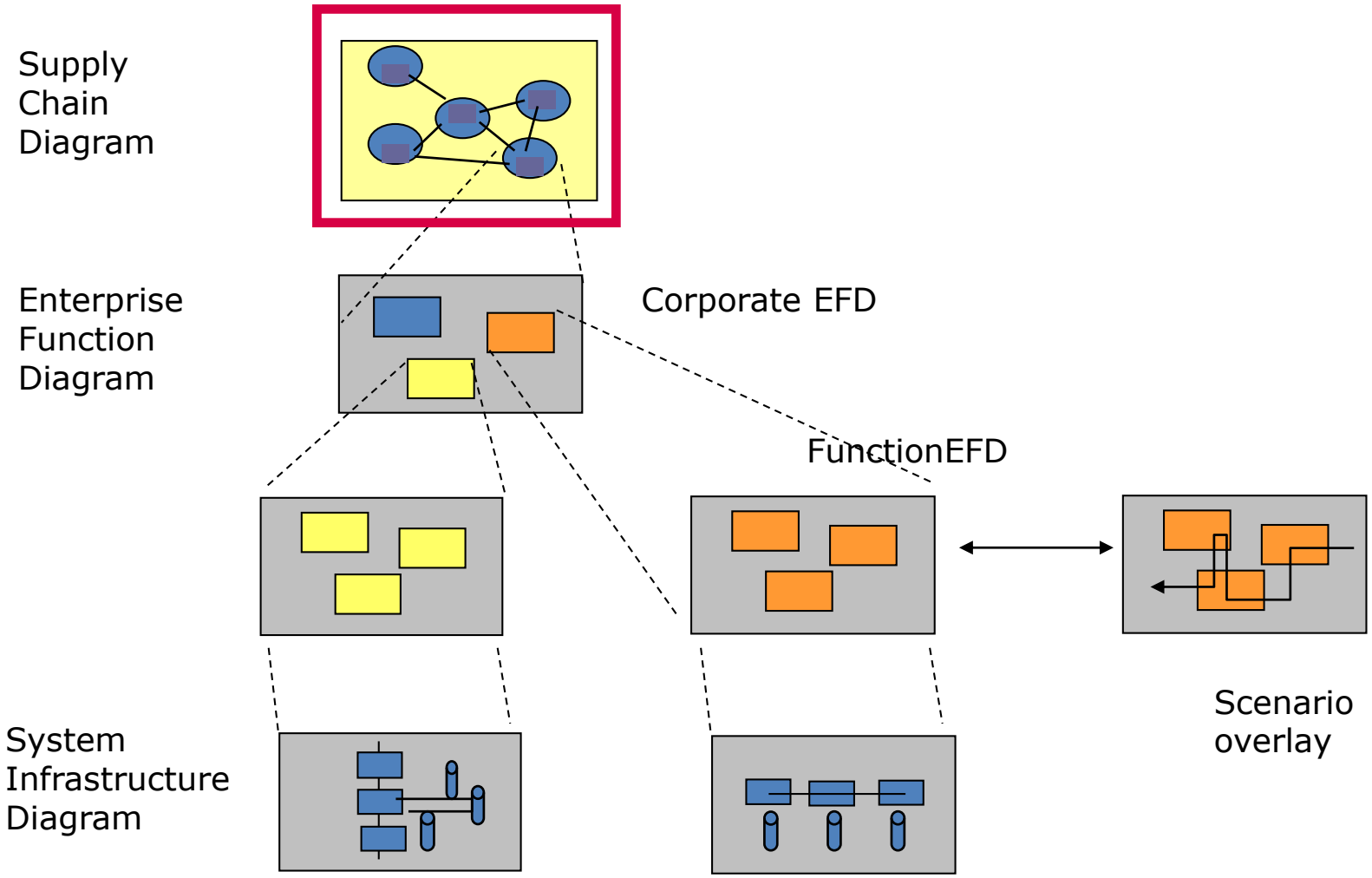
- Is the previous model a data model (M1) or meta-data model (M2)?



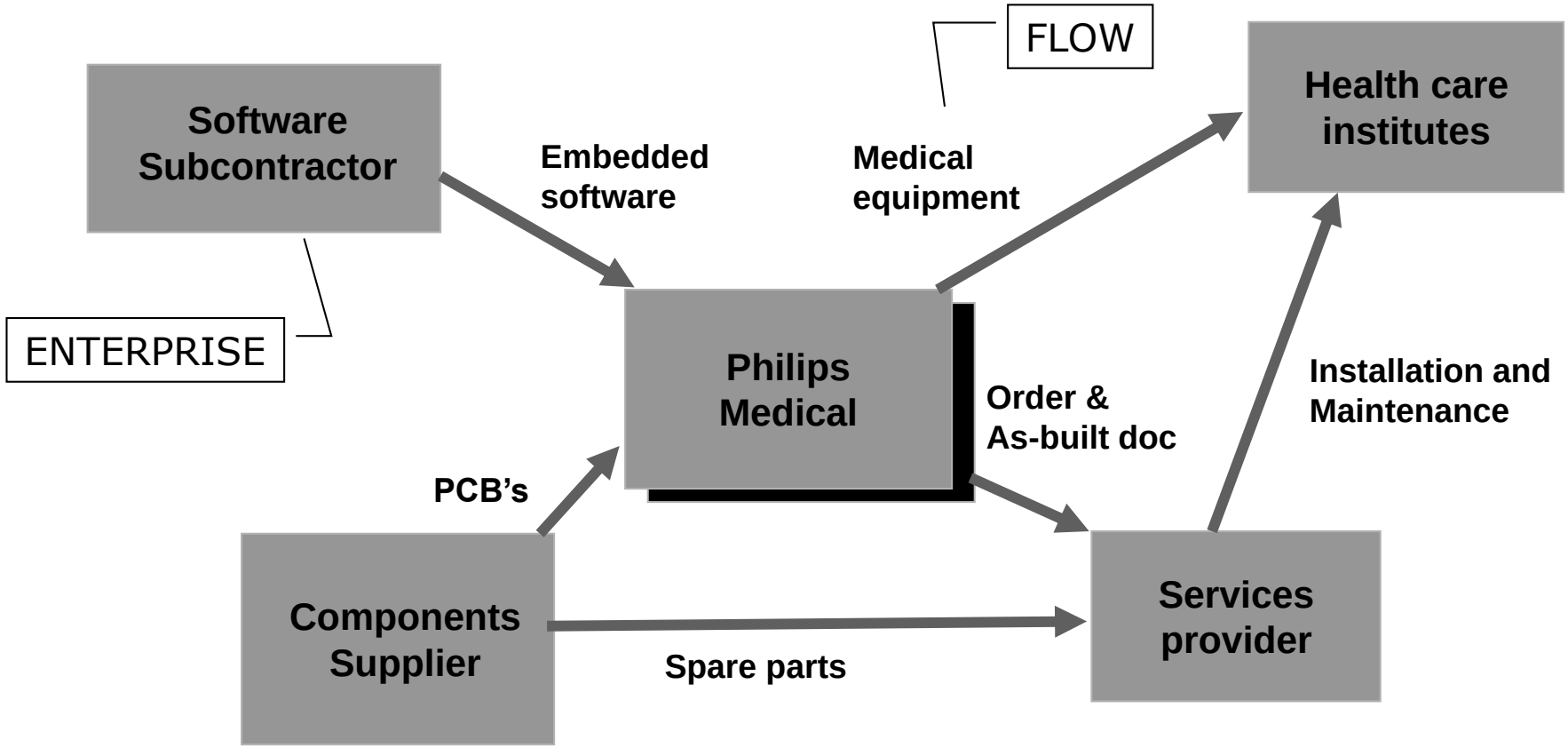
Supply chain diagram



Enterprise Architecture Models (EAM)

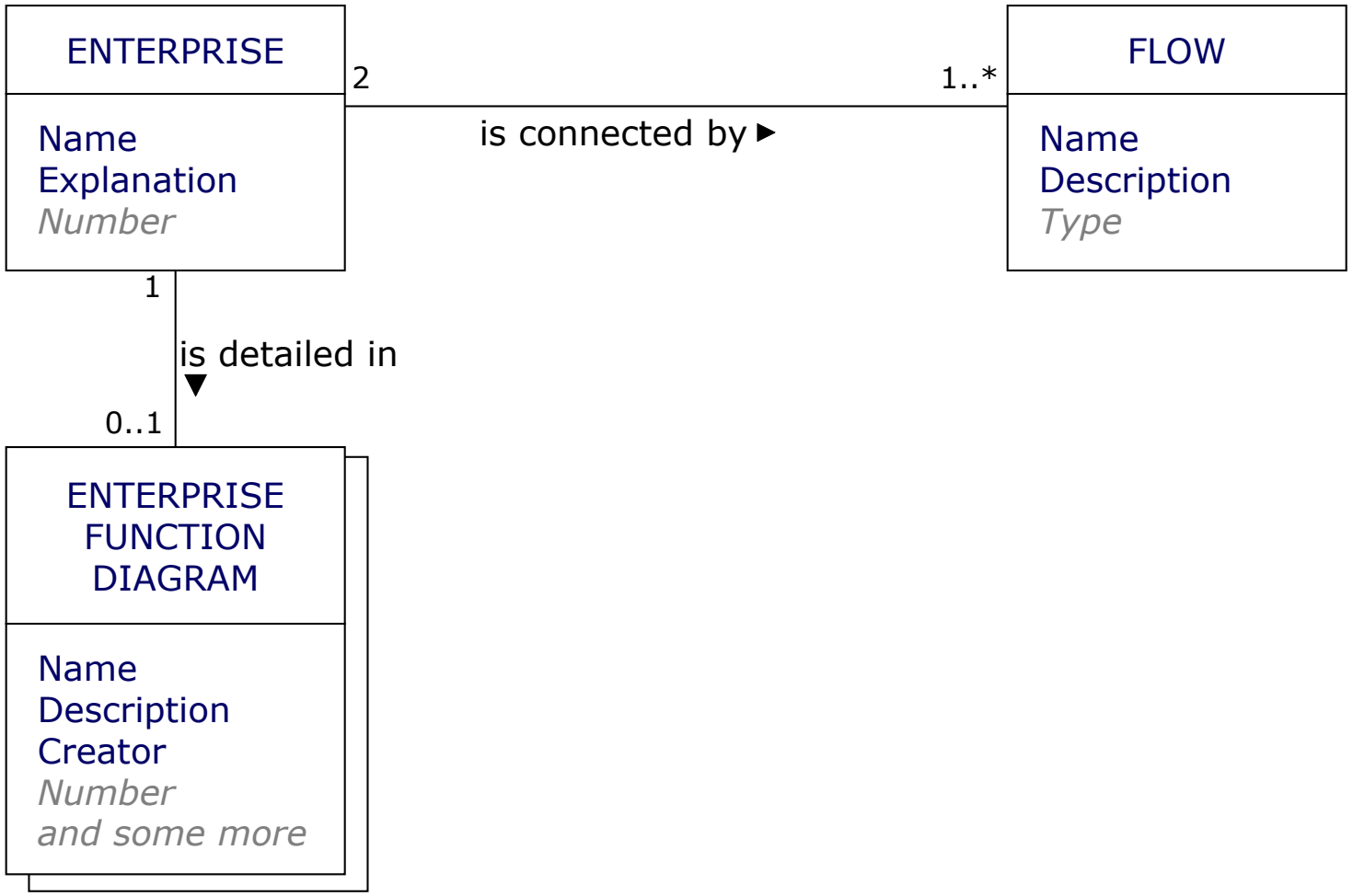


Supply chain diagram



Link to EFD of central enterprise; expressed by shadowed box

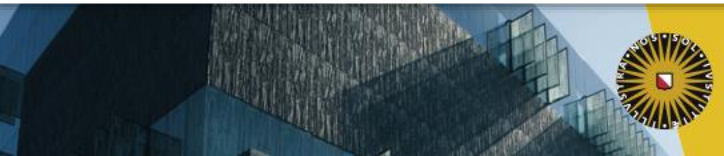
Meta-model of SCD



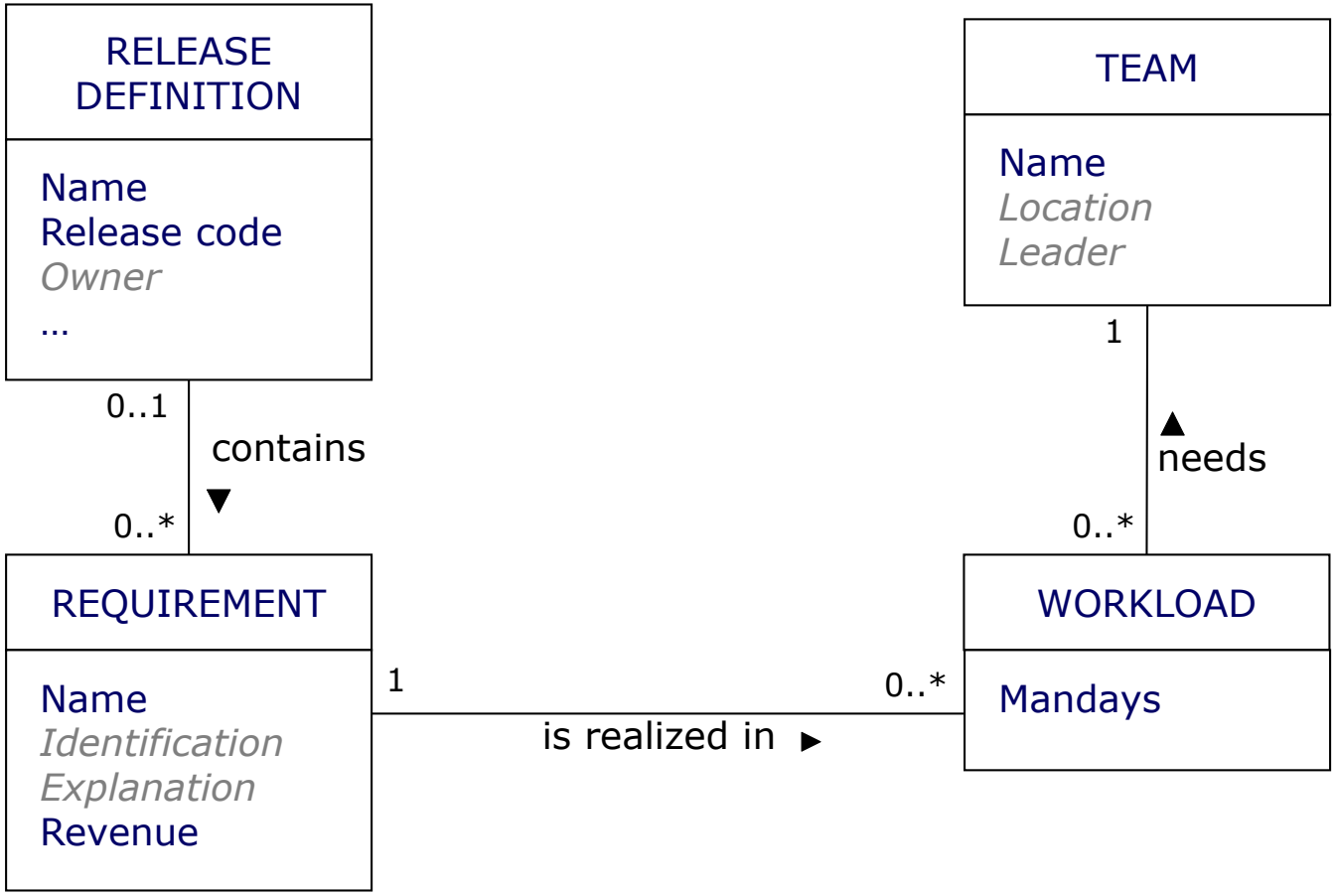
Assignment: Release definition

Release Definition 3.1					
Requirement	**Total**	**Team A**	**Team B**	**Team C**	**Revenues**
Autorisation on order cancellation and removal	50	5		45	24
Autorisation on archiving service orders	10	2	5	5	12
Performance improvements	15	15			20
Inclusion grafical plan board	70	10	10	50	100
Link with Acrobat reader for PDF files	33		33		10
Optimalising interface with international Postal code system	15			15	10
Adaptations in rental and systems	40		20	20	35
Adaptations in symbol import	10	10			5
Comparison of services per department	34		9	25	10
Totals	**277**	**42**	**77**	**160**	**226**

What does the meta-data model look like?

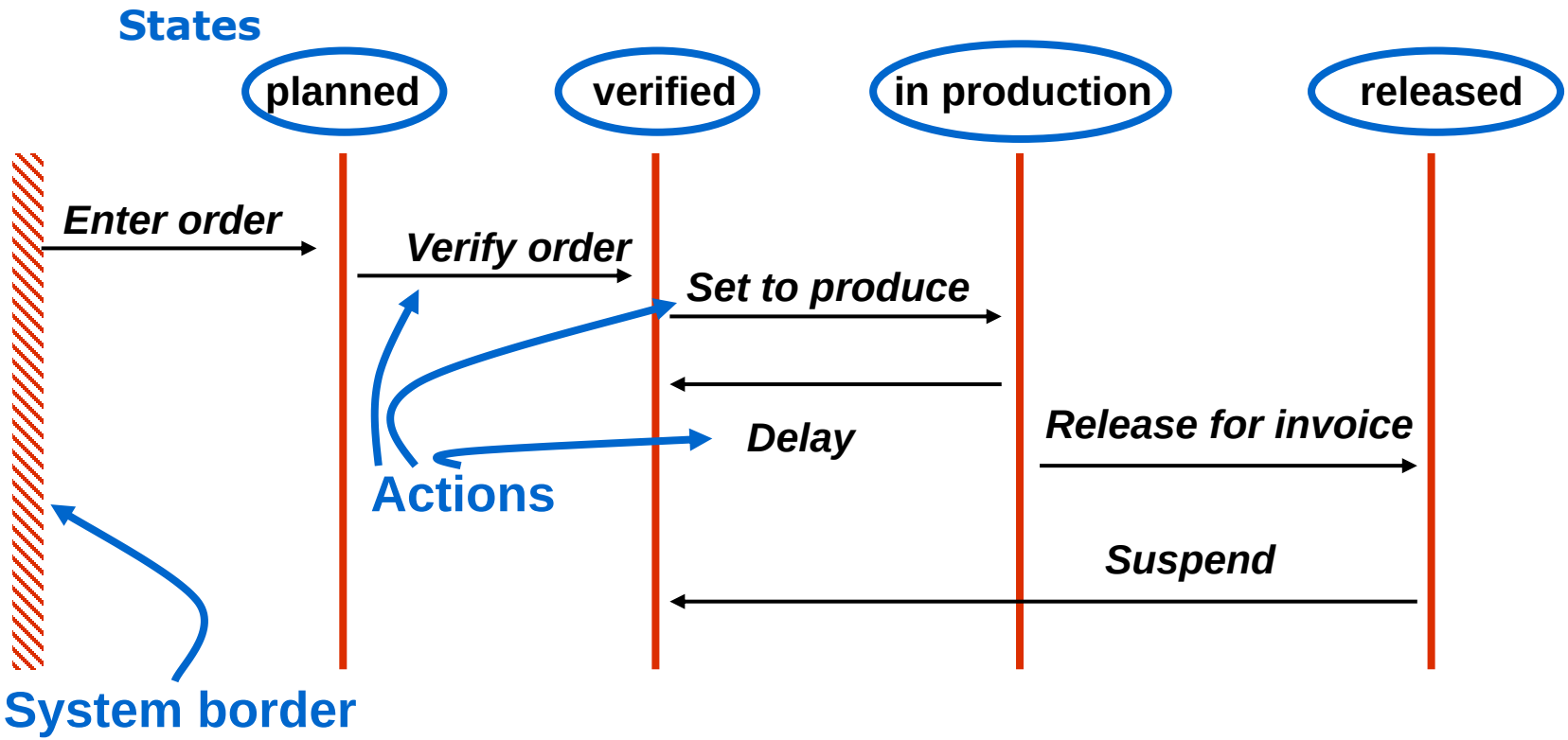


Meta-model of release definition

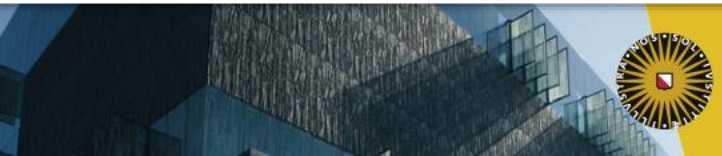


Assignment: sequence diagram

Create the meta-model for the Sequence diagram



QUESTIONS?



References

Bruegge, B., & Dutoit, A. H. (2003). *Object-oriented Software Engineering: Using UML, Patterns and Java.* Upper Saddle River, NJ: Prentice Hall.

Goldfine, A., & Konig, P. (1985). A technical overview of the information resource dictionary system. *Computers and Standards 4*(3), 153-208.

OMG (2006). *Meta Object Facility (MOF) Core Specification*, version 2.0,. document formal/06-01-01. Retrieved February 13, 2011, from http://www.omg.org/spec/MOF/2.0/PDF/

van de Weerd, I., & Brinkkemper, S. (2008). Meta-modeling for situational analysis and design methods. In M.R. Syed and S.N. Syed (Eds.), *Handbook of Research on Modern Systems Analysis and Design Technologies and Applications* (pp. 38-58). Hershey: Idea Group Publishing.

Inge van de Weerd, Sjaak Brinkkemper, Johan Versendaal: Incremental method evolution in global software product management: A retrospective case study. Information & Software Technology 52(7): 720-732 (2010)