Meta-Modeling and Modeling Languages

Prof. Dr. Knut Hinkelmann

Models and Modelling

Model

A reproduction of the part of reality which contains the essential aspects to be investigated.

Modelling

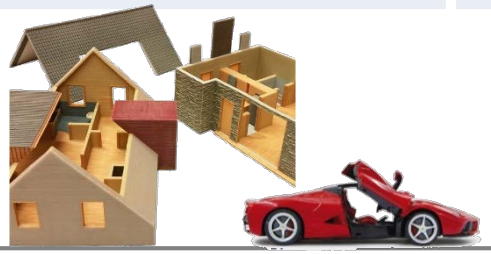
Describing and representing all relevant aspects of a domain in a defined ***modeling language***.

Result of modelling is a model.

Models and Modelling Language

- There can be different kinds of models
- A modelling "language" specifies the building blocks (elements) from which a model can be made.
- There can be different types of modelling languages, depending on the kind of model

Kind of model	Examples	Modeling elements
graphical model	BPMN, Archimate	boxes and arrows
textual description	English, HTML	words
mathematical model	formulas	operators, numbers, symbols
physical model	Architecture model	Objects, connectors



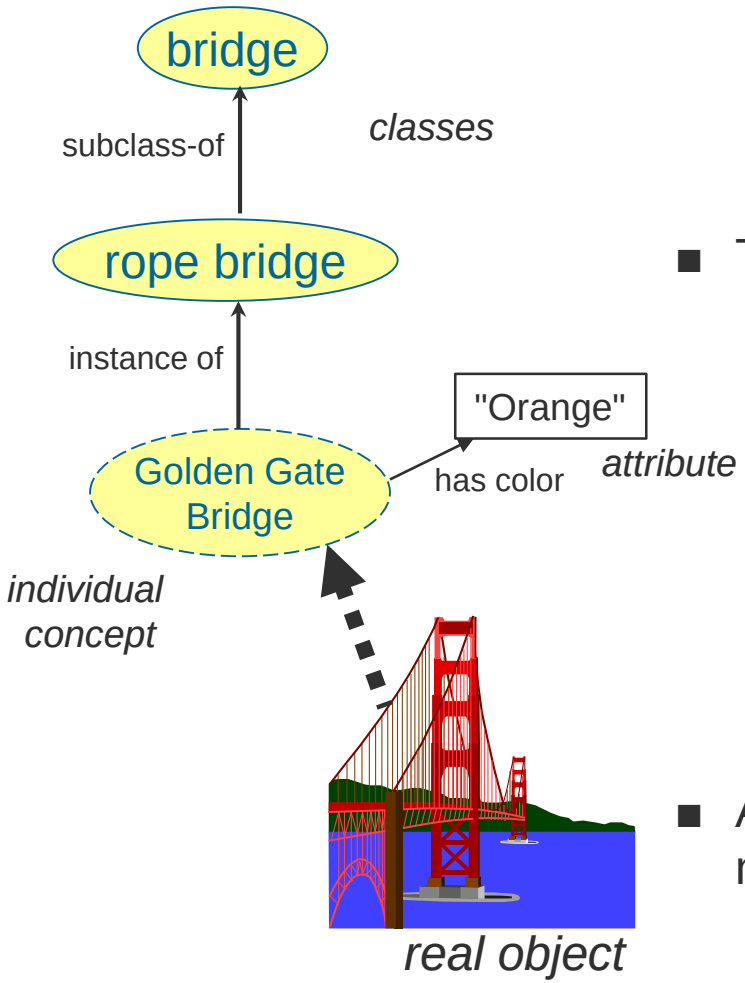
Domain-specific vs. General-purpose Modeling Languages

- General-purpose modeling languages can be used to represent any kind of knowledge
- Domain-specific languages are notations which are defined to model knowledge about a specific domain

General-purpose Modeling Languages

- General-purpose modeling languages can be used to represent any kind of knowledge
- There are a wide range of general-purpose modeling languages
  - Natural language allows to express any knowledge
  - Formal languages: Typically a subset of Logic
  - Graphical Diagrams
- General-purpose graphical modeling languages have been developed in a many difference fields:
  - Artificial Intelligence: Semantic networks, Ontologies
  - Data Modeling: Entity Relationship Diagrams
  - Object-Oriented Programming: UML Class Diagrams

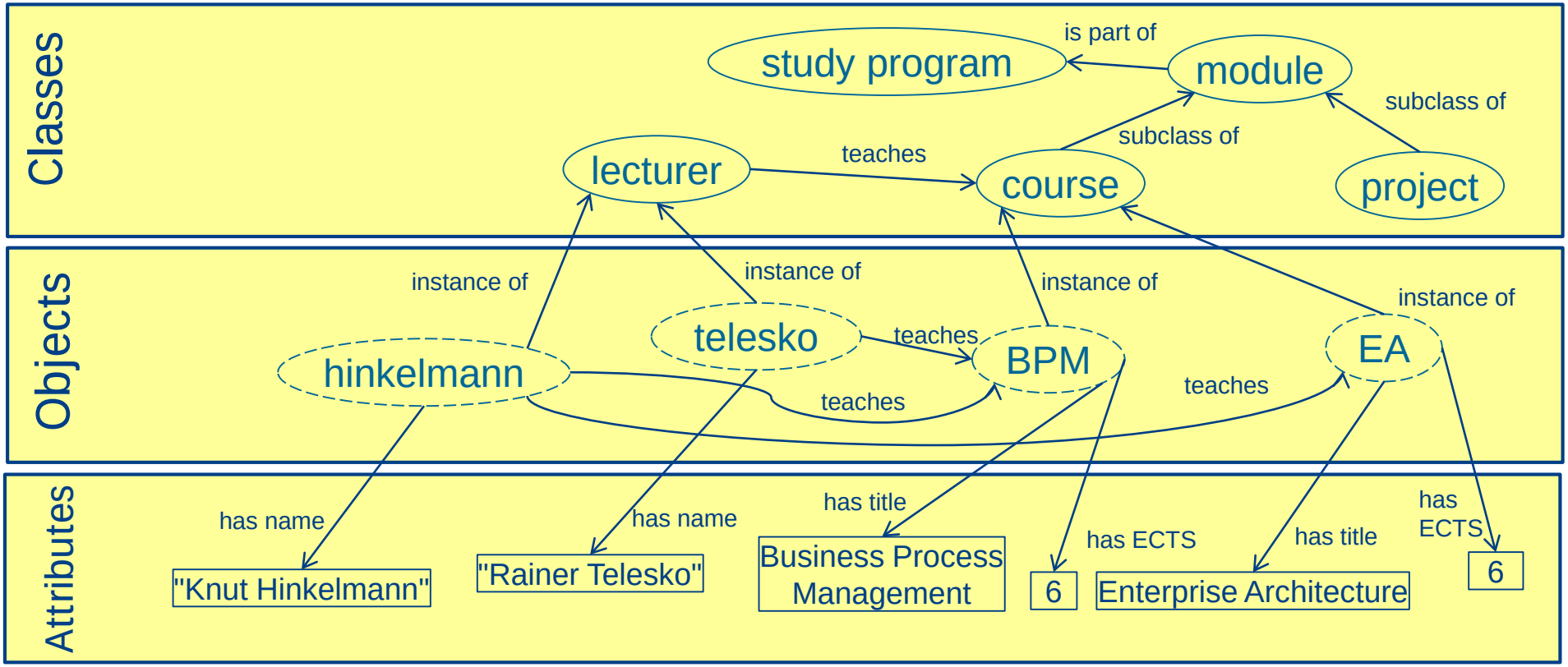
General-purpose Modeling: Concepts and Relations



- There are two kinds of concepts:
  - **classes** (also called general concepts)
  - **objects** (alsoc alled individual concepts, individuals or instances)
- There are different kinds of relations
  - **generalisation** ("is a")
    - between classes **(subclass of)**
    - between object and class **(instance of)**
  - **aggregation and composition**
    - "part-of" relationship
  - **associations**
    - any other kind of relationship
- Attributes can be regarded as associations whose value is not node but is of a primitive type (number, string).

Modeling with a General-purpose Modeling Language

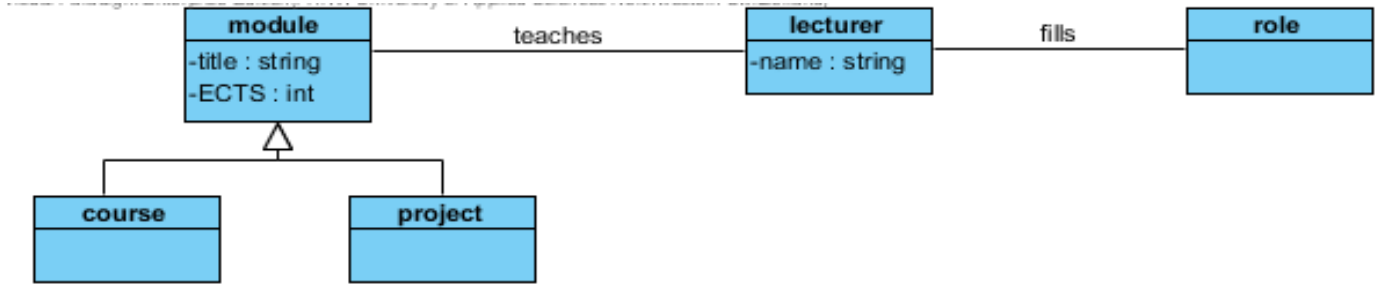
With a general-purpose modeling language, knowledge of any domain can be modeled. This is a model for modules of a study program.



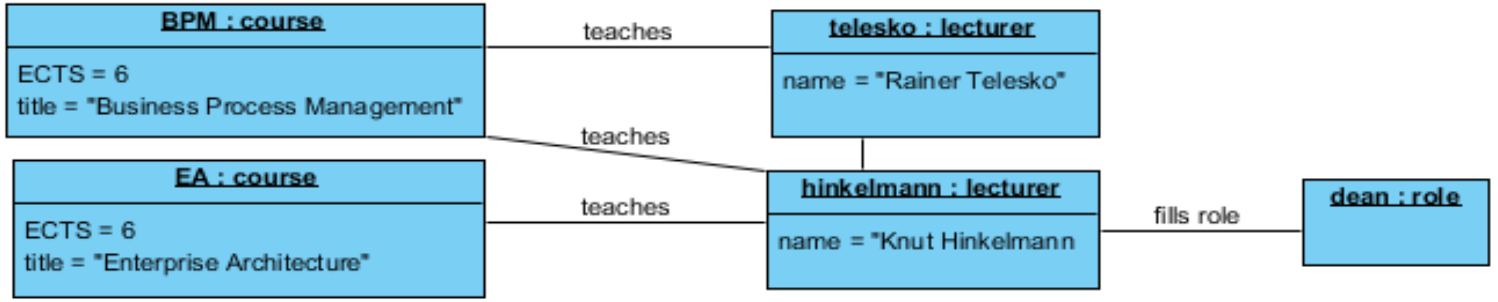
Modeling with a General-purpose Modeling Language

The metamodel for this generic modeling language corresponds to subsets of UML Class Diagrams and UML Object Diagrams

Classes



Objects



The classes specify a (new) domain-specific metamodel – In this case for modeling modules of a study program

Disadvantage: No specific modeling shapes

Strengths and Weaknesses of Domain-specfic Modeling Languages

- Strengths
  - Applicability
    - Can be used to represent everything
    - Every model in the same language
    - Low learning curve for the language
- Weakness
  - No guidance: Users have to …
    - determine how to structure a domain
    - to identify relevant concepts
  - Restrictred reusability
    - Different applications use different concepts

Domain-specific Modeling Languages

- Domain-specific modeling languages have modeling elements for typical concepts and relations of a domain of discourse
- Examples of domain-specific modeling languages:
  - **BPMN** is a domain-specific language for business processes
    - Modeling elements: task, event, gateway, ….
    - relations: sequence flow, message flow, data association, …
  - **ArchiMate** is a domain-specific language for enterprise architectures
    - Modeling elements: process, actor, role, business object, …
    - relations: uses, realizes, …

Strengths and Weaknesses of Domain-specfic Modeling Languages

- Strengths
  - Comprehensiblity of models
    - elements and relations are adequate for stakeholders
    - domain-specific shapes
  - Standardisation: Reuse of models
    - Common concepts for a domain (e.g. BPMN, ArchiMate)
- Weaknesses
  - Restricted to a specific domain
    - Only what can be expressed with the modeling elements can be modeled

What do we do if there is no Domain-specific Modelling Language

- If there is no appropirate domain-specific modelling language for a domain of interest, we have two options
  1. Use a general-purpose modeling language
  2. Define a new domain-specific modelling language
  3. Customization of an existing modeling language

Using a Domain-Specific Modeling Language

- Example: Modeling Data and Documents
- see chapter 7 – Implementing Enterprise Architecture

Customization in Archimate

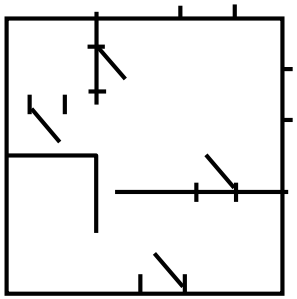
- The ArchiMate language contains only the elements and relationships that are necessary for general architecture modeling.
- It can be customized for more domain-specific by specialization of elements and relationship, e.g.
  - **Business Actor** could be
    - Individual
    - Organization Unit
  - **Product** could be
    - Physical Product
    - Digital Product
  - **Network** could be
    - WiFi Network
    - Wide Area Network
  - **Equipment** could be
    - Vehicle
    - Train

Model in Architecture

real object

house

model



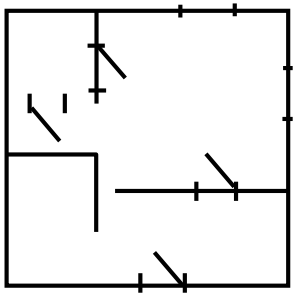
architect's drawing
(plan)

Model and Modeling Language in Architecture

real object

house

model

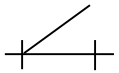


architect's drawing
(plan)

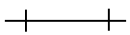
modeling language
(concrete syntax)

object types:

wall

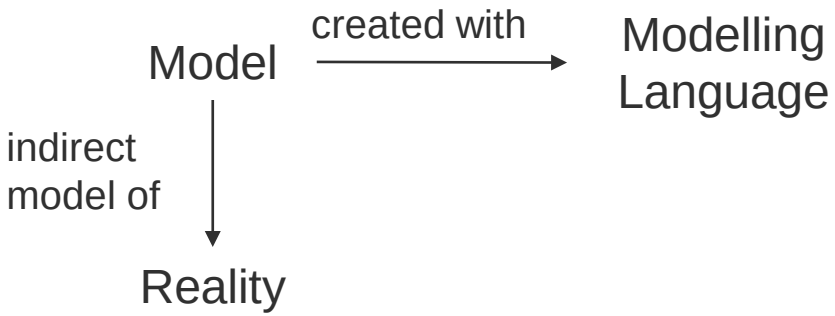


door



window

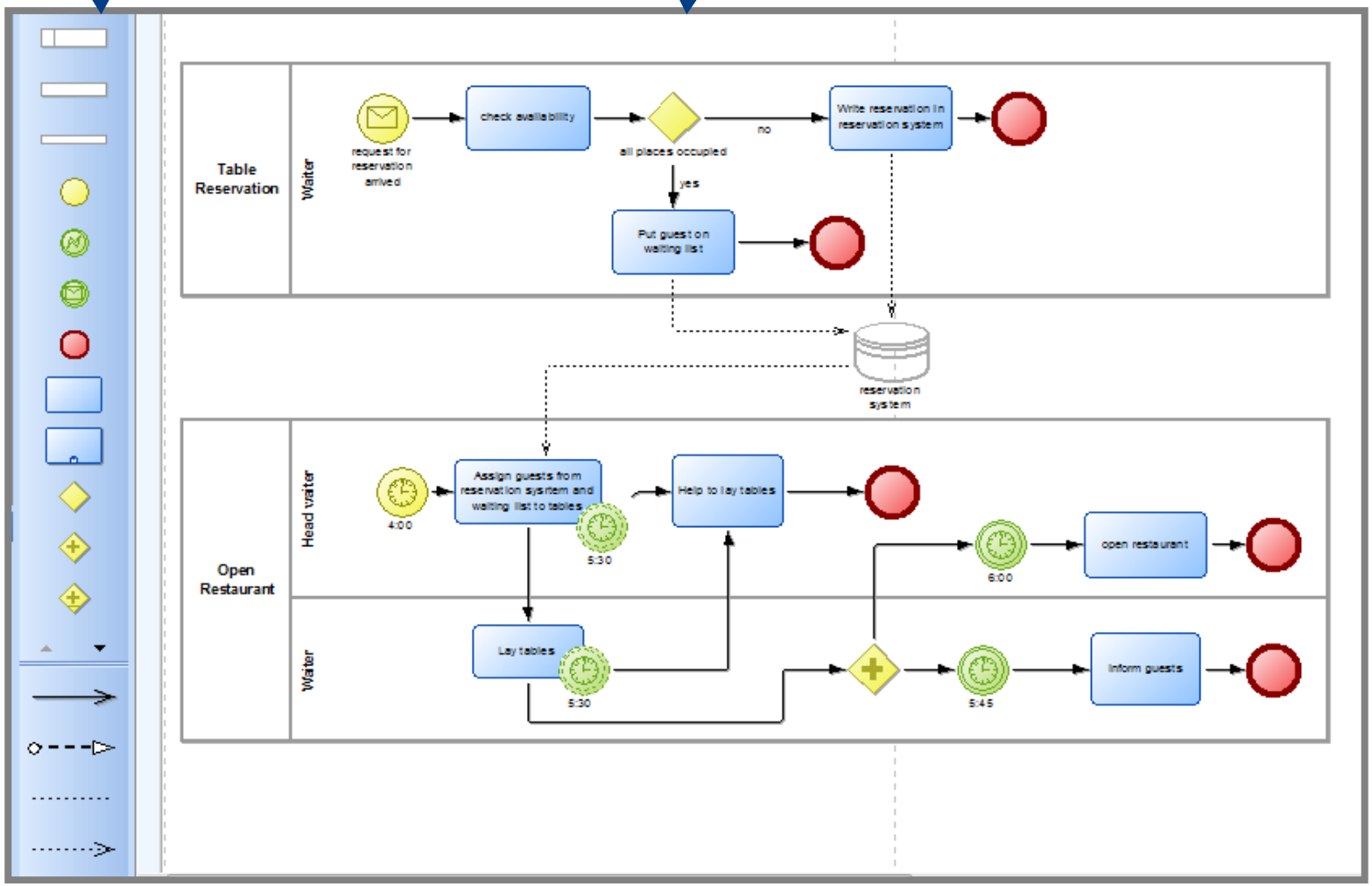
Modelling Language



- A modelling "language" specifies the building blocks (elements) from which a model can be made.
- There can be different types of modelling languages
  - graphical model
  - textual description
  - mathematical model
  - conceptual model

Modeling Language

Model

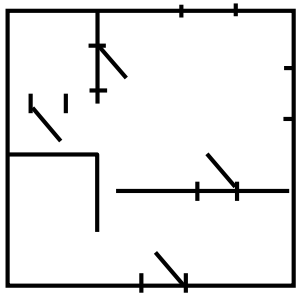


Model and Meta-Model in Architecture

real object

house

model

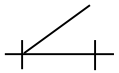


architect's drawing (plan)

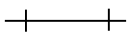
modeling language
(concrete syntax)

object types:

wall



door



window

meta-model
(abstract syntax)

object types:
- wall
- door
- window

rules:
- a door is adjacent to a wall on both sides
- Windows are on outer walls.

Metamodel and Modeling Language

Metamodel

- The *metamodel* is a model of a model. It defines the modeling elements (concepts, relations, constraints) without specifying the layout and notation

Modeling language

- The *modeling language* defines the notation/appearance of the modeling elements

Illustration: Meta-model and Model for Processes

Metamodel:

Abstract syntax:
Concepts which can be used to create models.

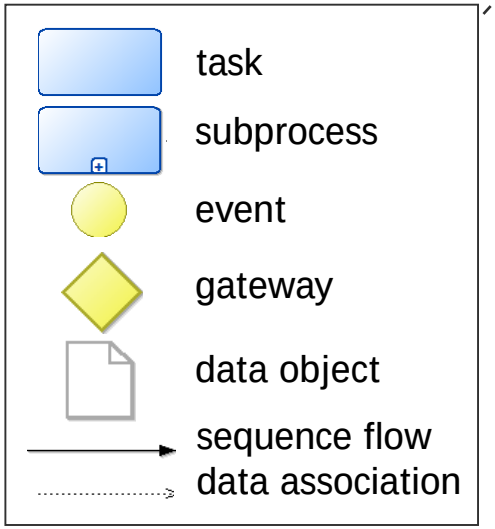
Example: A process model consists of concepts for

- «task», «subprocess», «event», «gateway», «data object»
- «sequence flow», «data association».

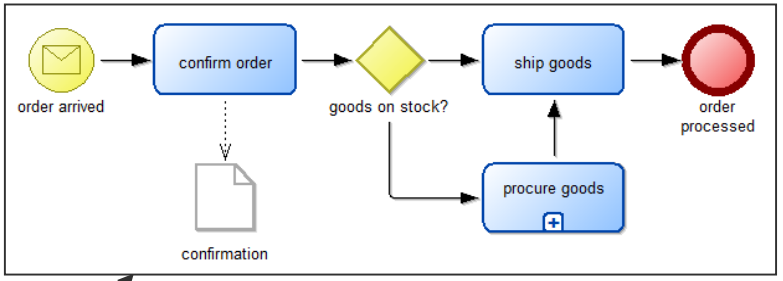
The elements have attributes and there are rules how the elements can be combined.

Modeling Language:

Concrete syntax:
Notation/appearance of meta-model elements

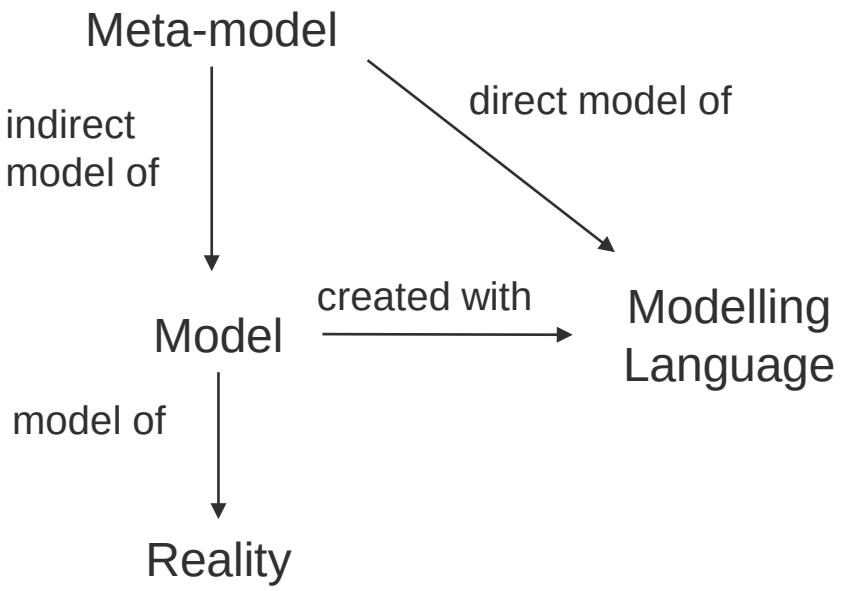


Model:



A model contains instances of the object types defined in the meta-model, according to the concrete syntax of the modeling language. The object „confirm order" represents a real entity; it is an instance of the object type «task"

Meta-model

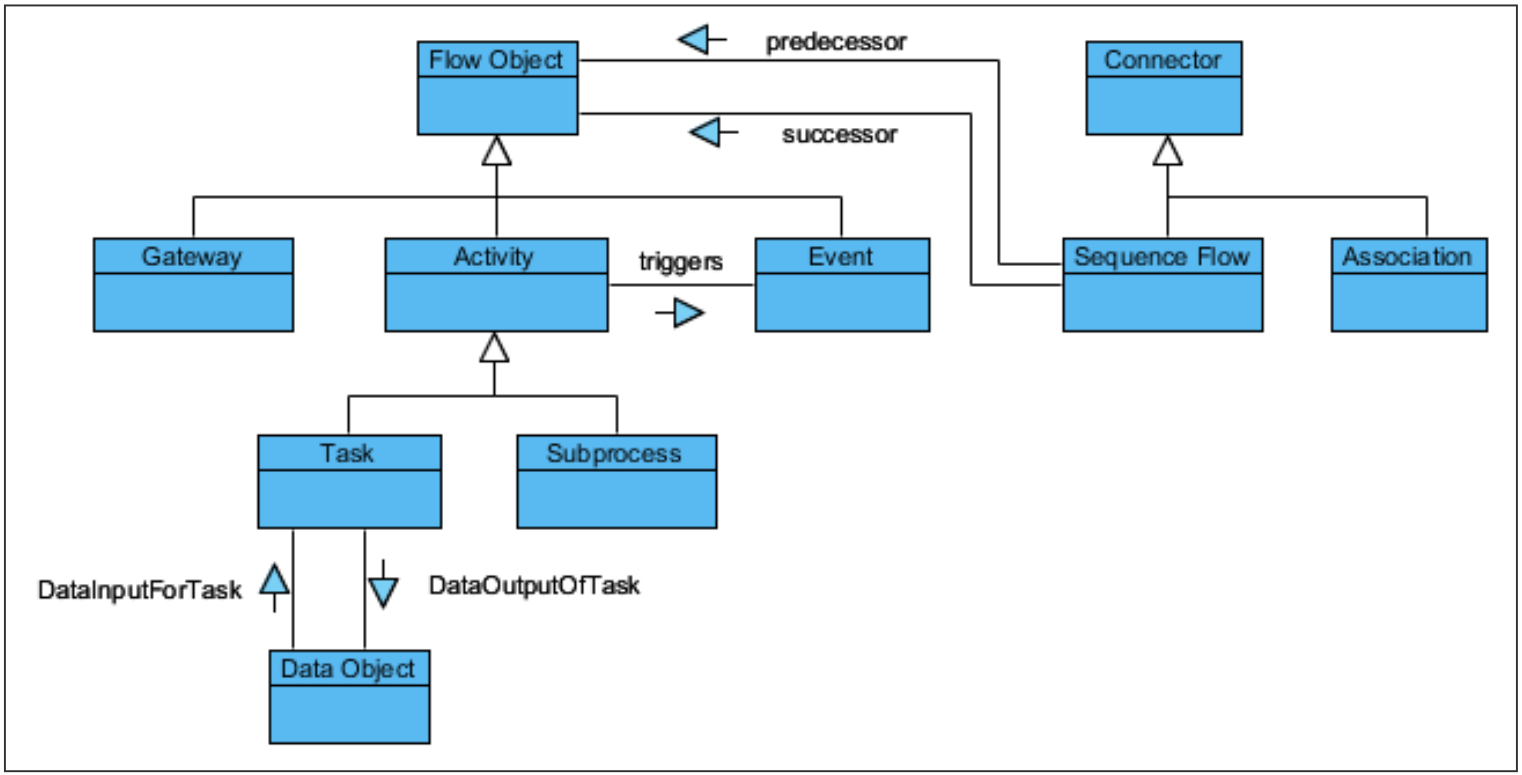


A meta-model defines the semantics of the modelling language, i.e. the building blocks that can be used to make a model. It defines the

- object types that can be used to represent a model
- relations between object types
- attributes of the object types
- rules to combine object types and relations

Metamodels can be defined as Class Diagrams

To model a metamodel one can use (a subset of) UML class diagrams

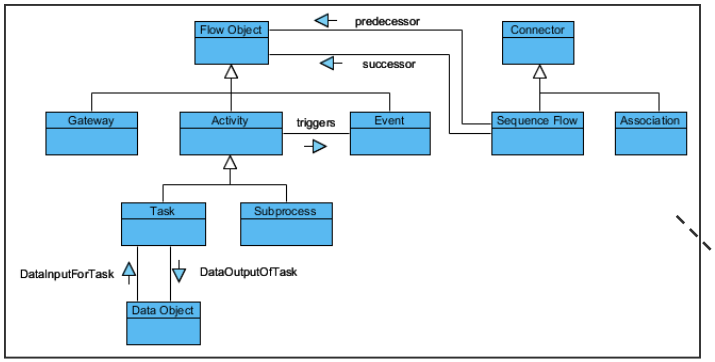


(UML Class diagrams where originally designed for modeling in object-oriented programming. This is why they contain operations and other features, which are not relevant for most modeling languages)

A Domain-specific Metamodel for Processes

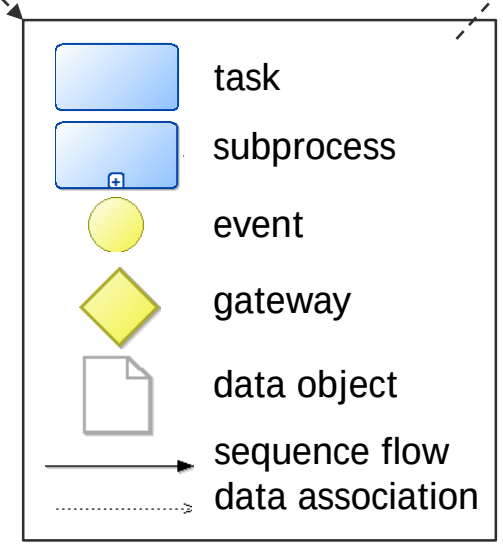
Meta-model:

- Classes and relations that can be used for modeling

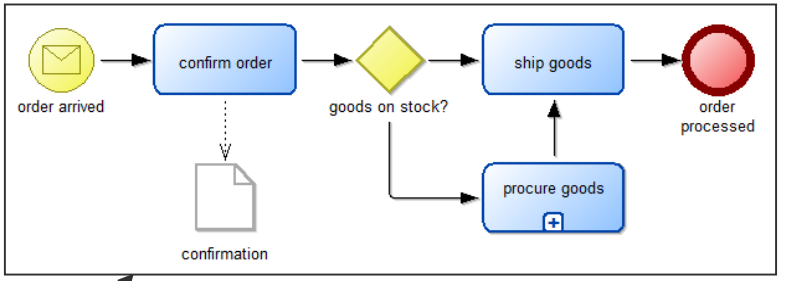


Modeling Language:

Concrete Syntax (notation, appearance) of meta-model elements

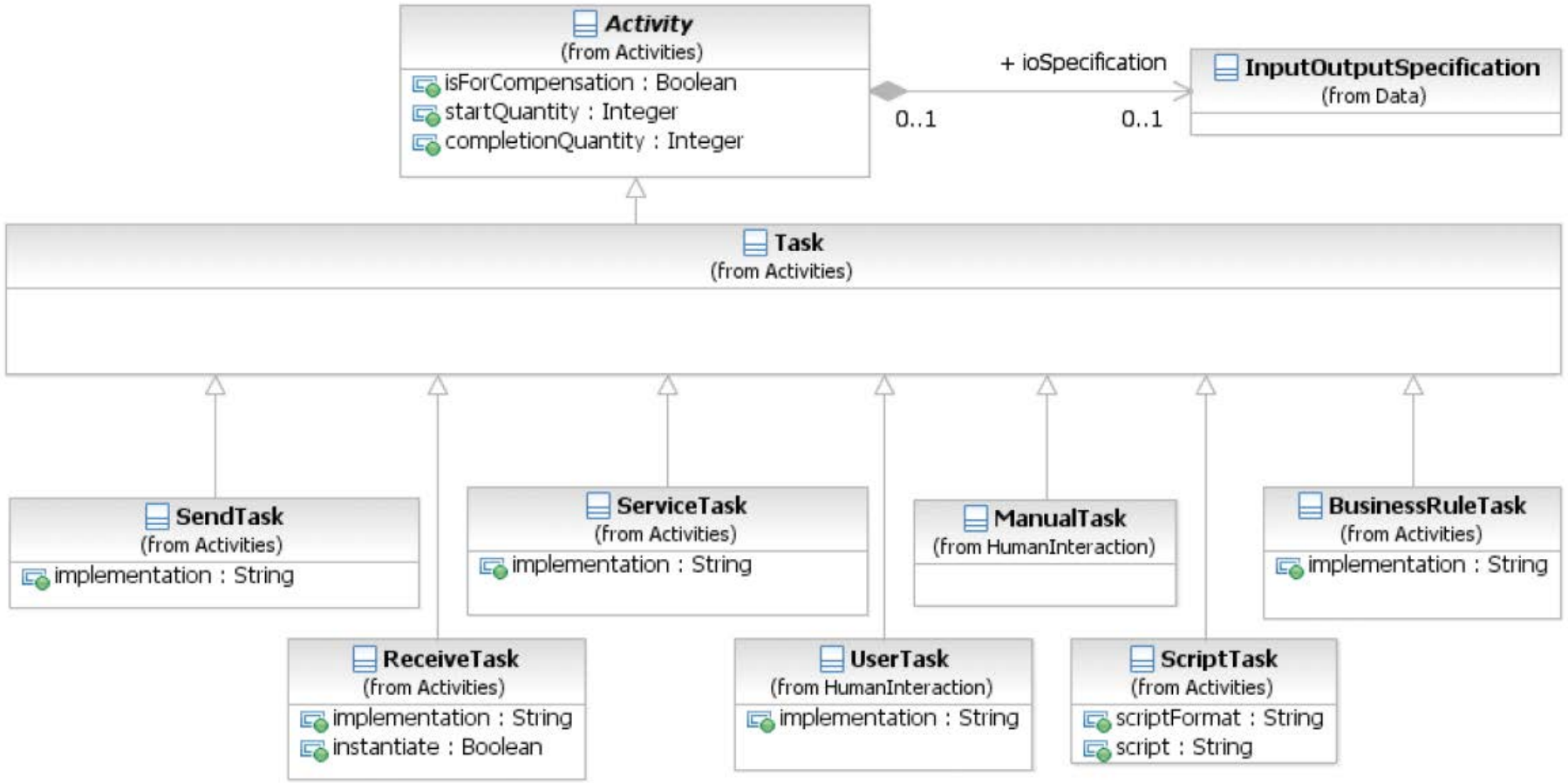


Model:



A model contains instances of the object types defined in the meta-model, according to the concrete syntax of the modeling language. The object „confirm order“ represents a real entity; it is an instance of the object type «task"

Subset of the BPMN Metamodel in UML



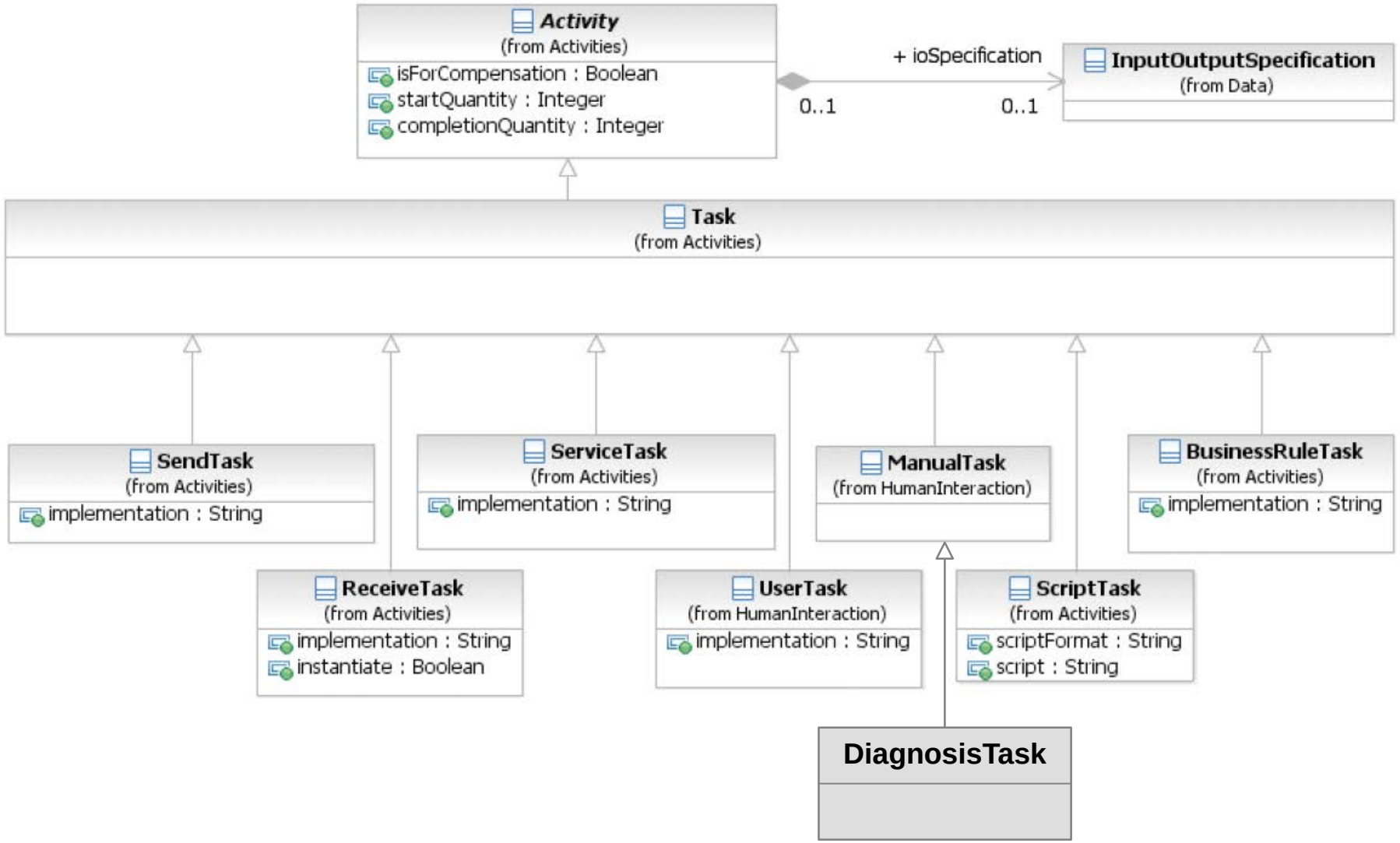
Source: BPMN 2.0 specification

Customization of BPMN

- BPMN is a domain-specific modeling language for business processes
- It would be possible to make BPMN more domain-specific for business processes in a specific application area, e.g.
  - Education: specific tasks for teaching like lecture, self-study, exam with predefined roles for lecturers and students
  - Health: specific tasks for diagnosis, therapy with roles like physician and patient

Customization of BPMN

In this example, BPMN is customization by adding a new subclass to the class ManualTasl



A Domain-specific Modeling Language can be regarded as a Customization of a General-purpose Modeling Language

- Example: In the Visual Paradigm tool we can use stereotypes to specialize UML class diagrams.
- We can define a new stereotype for a class and
  - change color
  - add an icon
- Example: stereotypes for modules and lecturer