Conceptual Modeling

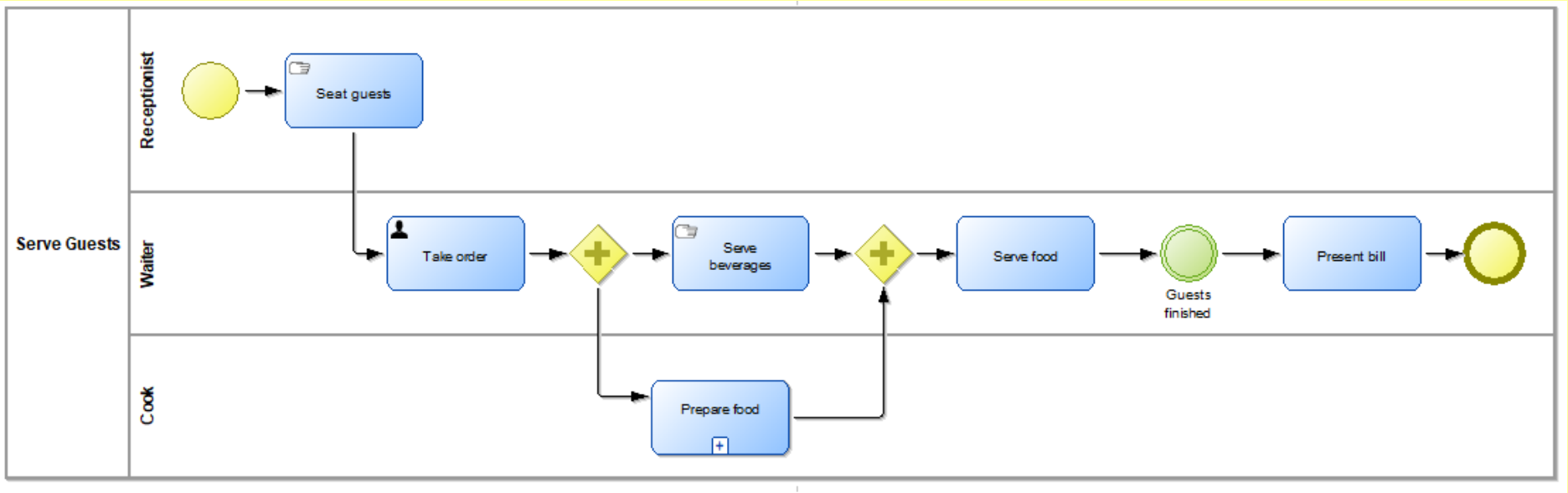
Knut Hinkelmann

Image by Bessi from Pixabay

Visual Communication

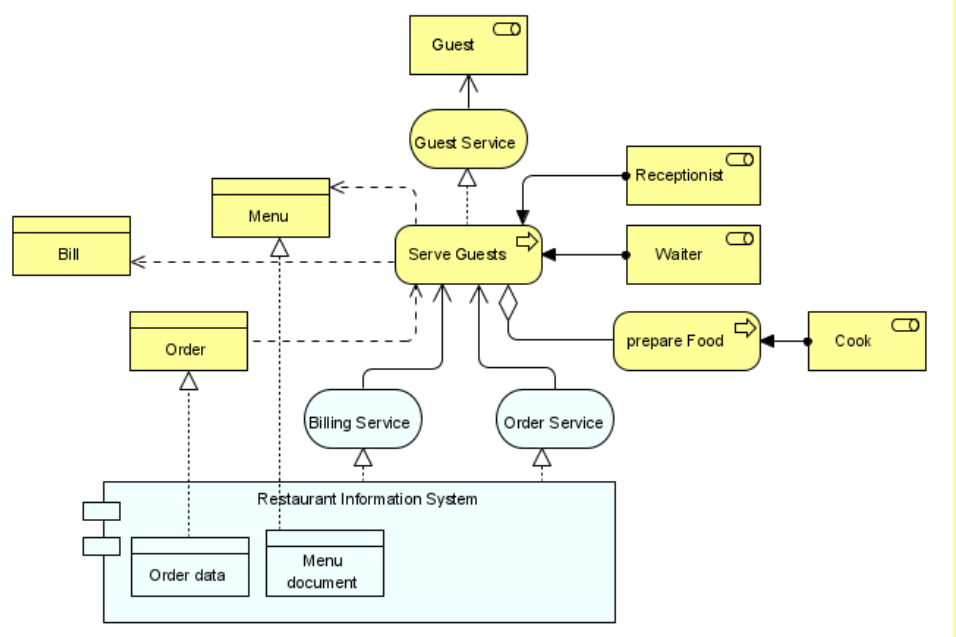
- A picture is worth a thousand words
- Graphical Models are easier to understand than text

Example: Interpretation of a Process Model



- Who seats the guests?
- Which task is executed after «Serve food»?
- Which activities are executed in parallel?

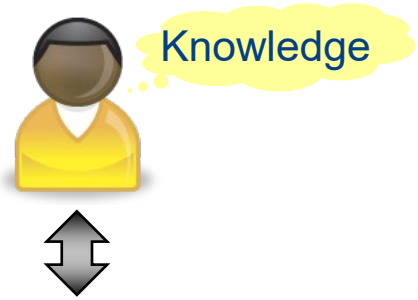
Example: Interpretation of an Architecture Model



- Who is involved in the process «Serve Guests»?
- Which Business Processes are served by the Restaurant Information System?
- In which application is the order data stored?

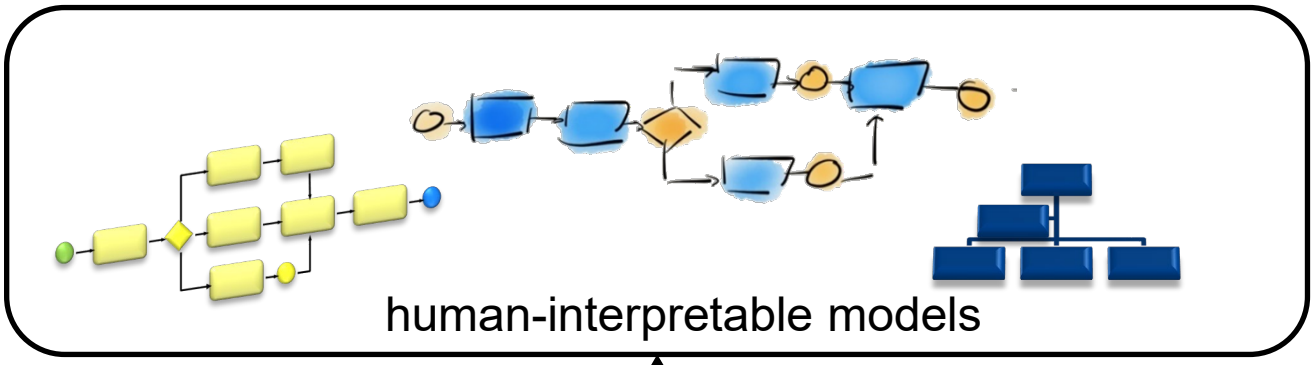
Graphical Models are appropriate for Humans

Communication/
Analysis/
Decision Making



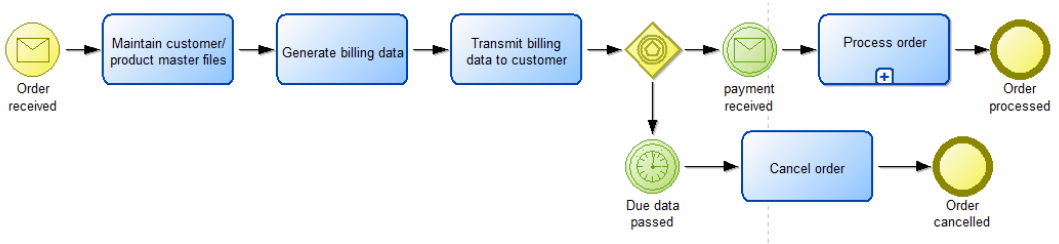
Examples:
Visio
Powerpoint

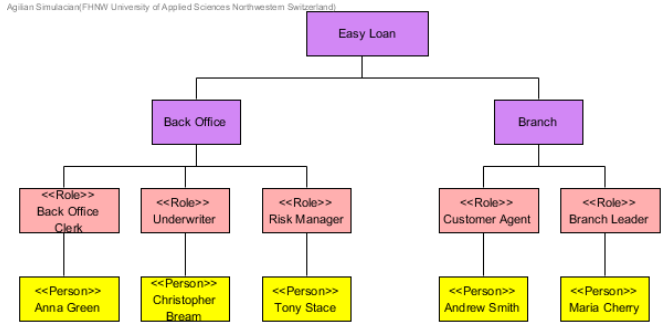
Models

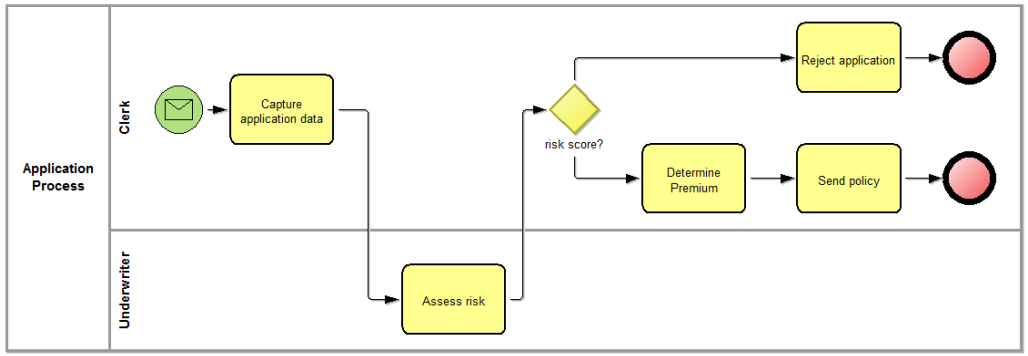


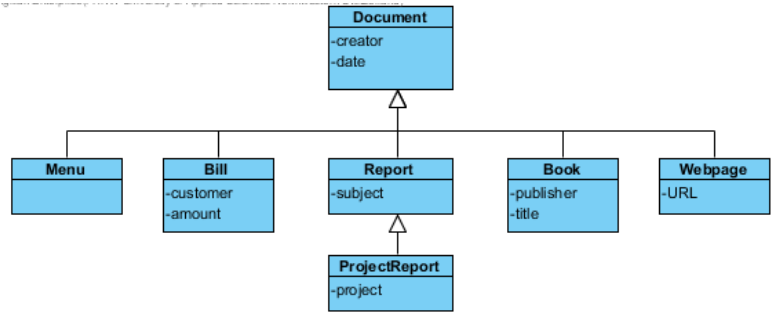
Reality

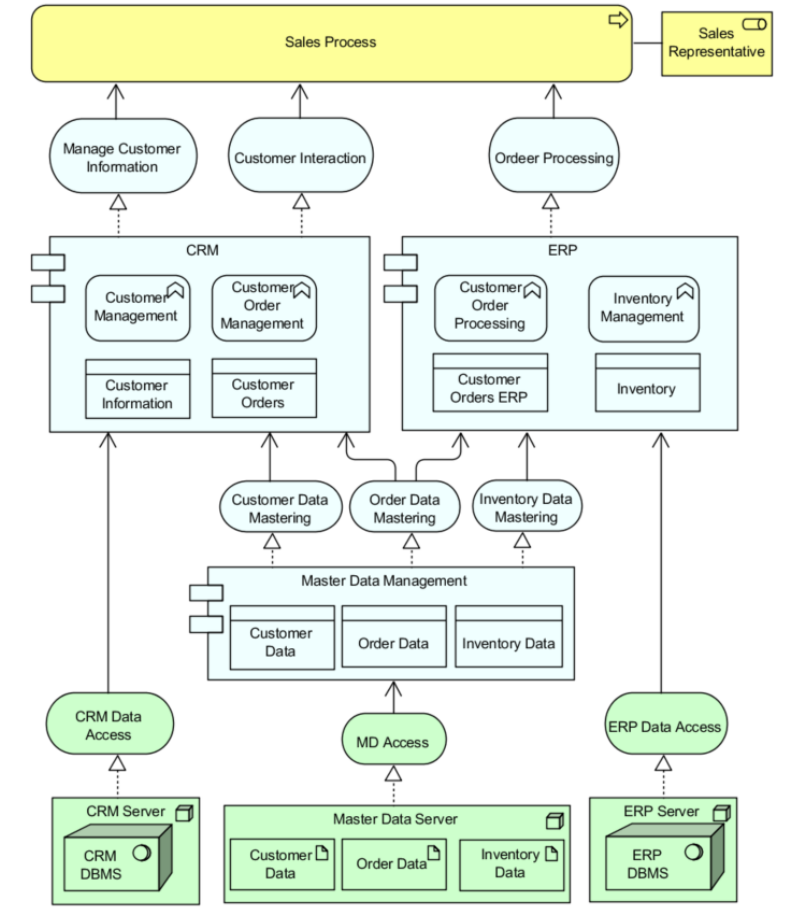
Enterprise Models



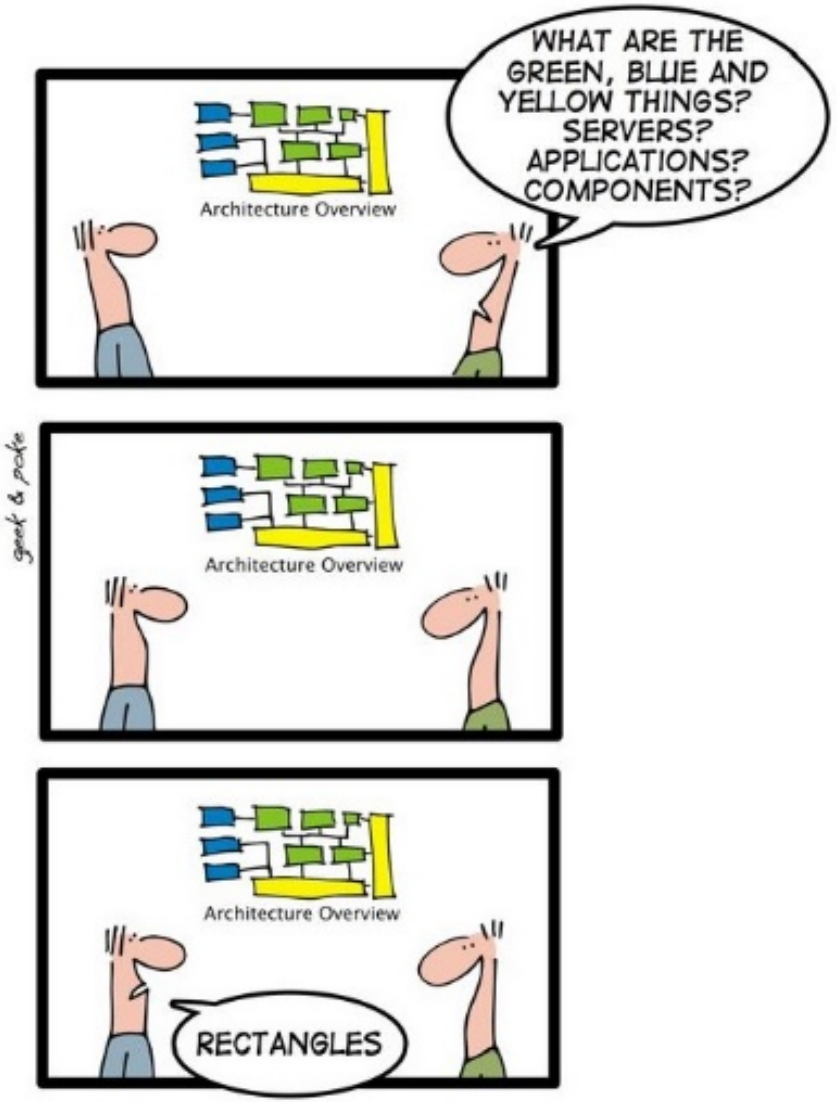








Interpretation of Models



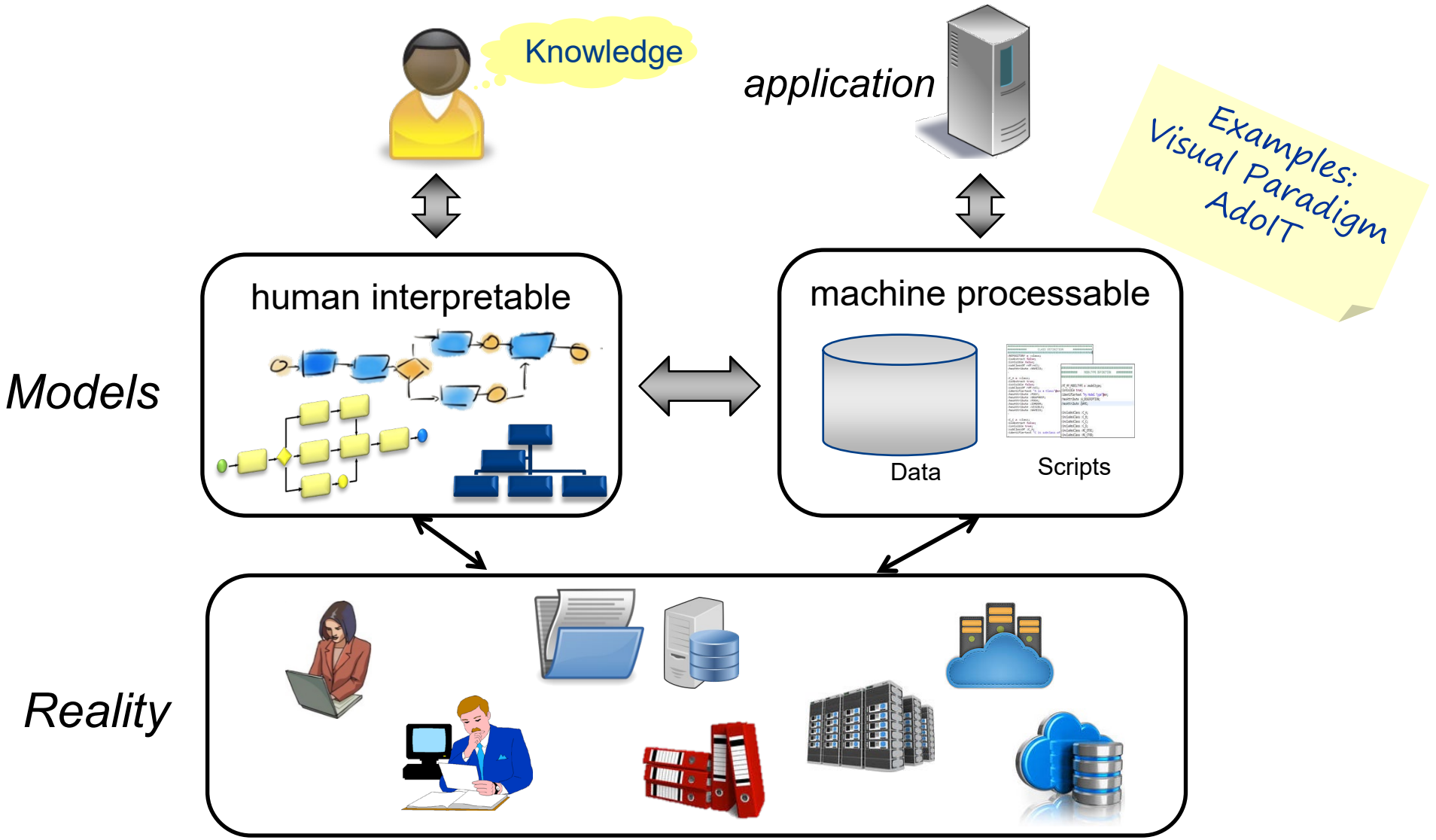
Models

- Models are not mere pictures; rather, they
  - provide a precise, meaningful description that can be visualized in different ways for different stakeholders;
  - can also be used to analyze the impact of changes, cost, risk, security, compliance and other relevant KPIs.

http://blog.bizzdesign.com/how-to-not-fail-when-implementing-strategy

Models should allow automated analysis, decision making and digitalization

Graphical Models are Represented in a Database



Conceptual Modeling and Metamodelling

Types of Modelling Languages

- ***General-purpose*** modelling languages can be used to represent any kind of knowledge
  - Examples:
    - Class diagrams (e.g. from UML)
    - Knowledge graphs (RDFS)
- ***Domain-specific*** languages have *predefined* concepts that are specific for a domain
  - Examples of domain-specific modelling languages:
    - BPMN: - Elements: task, event, gateway, ….
      - Relationships: sequence flow, message flow, association, …
    - ArchiMate: - Elements: process, actor, role, business object, …
      - Relationships: uses, realizes, …
- ***Conceptual modeling* =** using a domain-specific modeling language

Strengths and Weaknesses of Domain-specfic Modelling Languages

- Strengths
  - Comprehensibility of models
    - Concepts are adequate for stakeholders
  - Guidance for modelers
    - Predefined concepts determine what is relevant for a model
    - Modeling language determines correct usage of elements
  - 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 modelling elements can be modeled

Models, Modelling, Modeling Language

Model

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

Conceptual Modelling

Creating models using predefinded concepts.

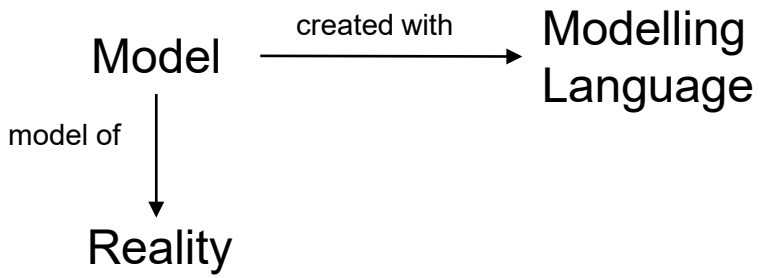
Meta Model

The concepts of the modeling language are predefined in a so-called meta-model

Modelling Language

Notation/Visualization of the concepts that can be used for modeling

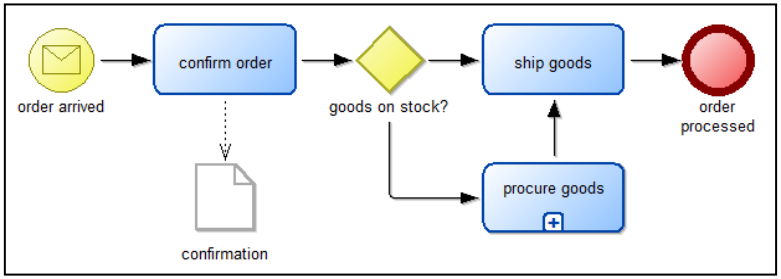
Modelling Language



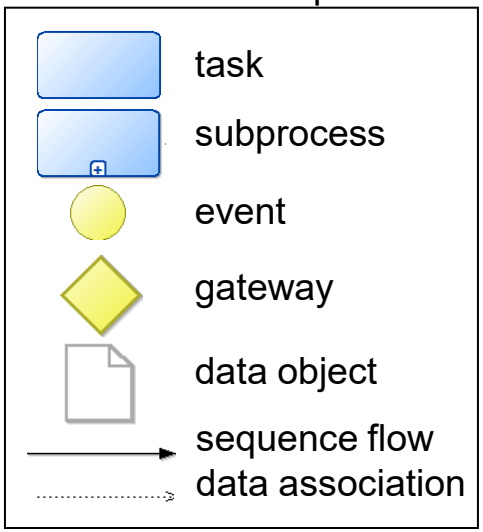
- A **modelling language** specifies the notation for the concepts, from which a model can be made.
- There are different kinds of notations
  - For graphical models the notation consists of *visualization* of the concepts
  - Textual models consist of words
  - Mathematical models use symbols
  - physical model are composed of physical elements

Illustration: Modeling Language for Business Processes

Model:



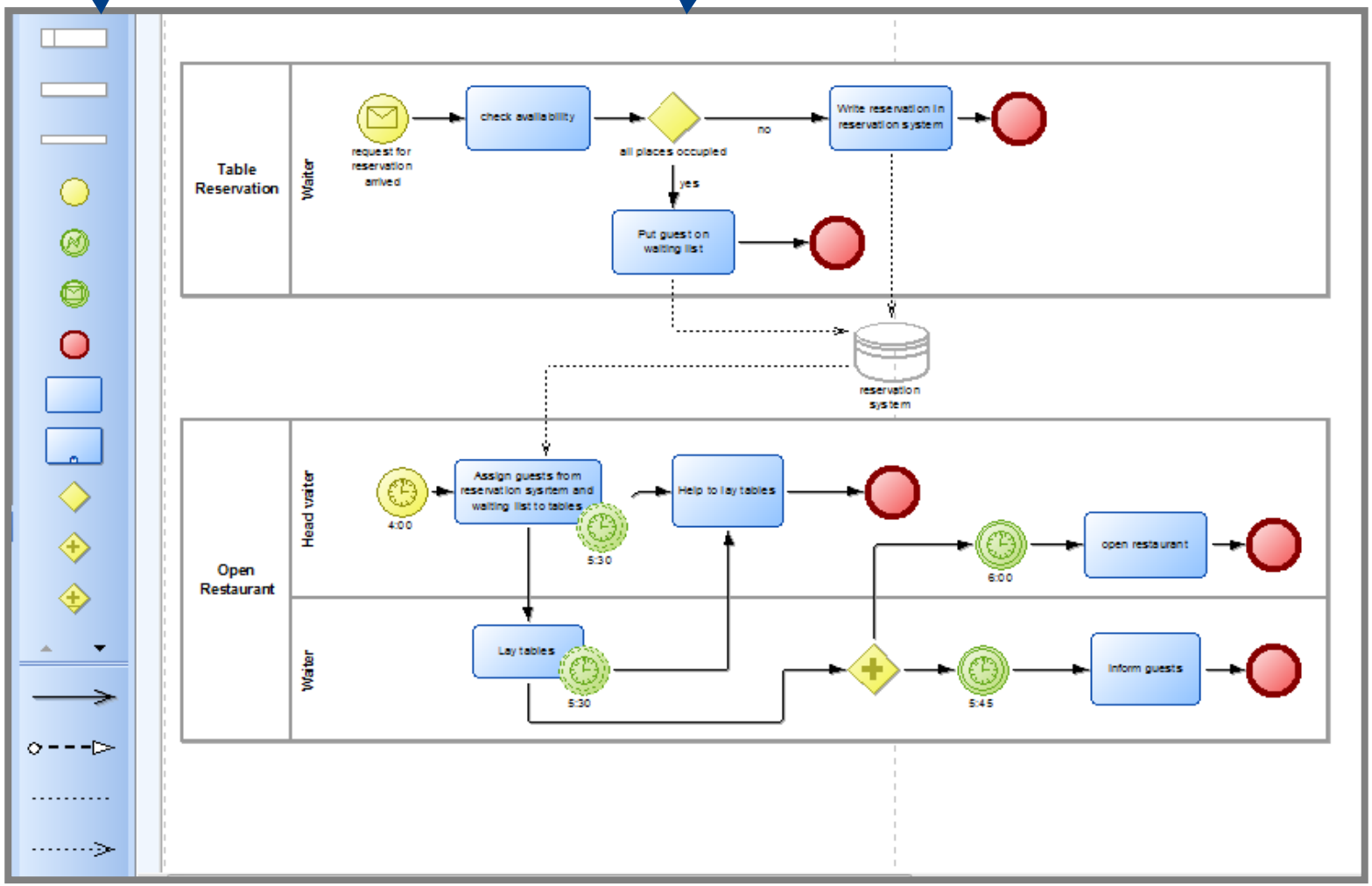
Modelling Language:
Notation/appearance of meta-model concept



A model contains instances of the concepts defined in the meta-model. The object „confirm order“ represents a real entity; it is an instance of the concept «task"

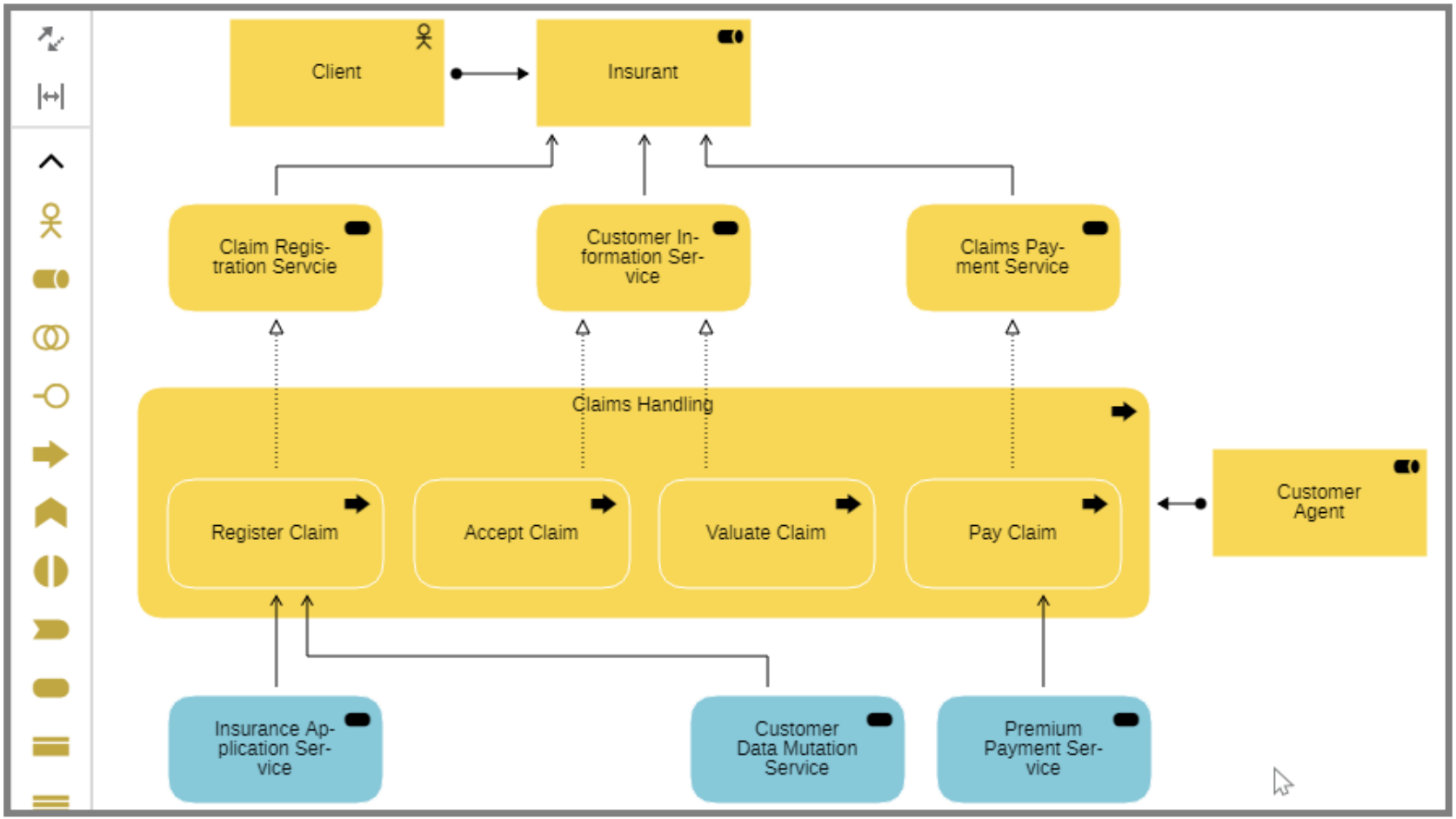
Modelling Language

Model

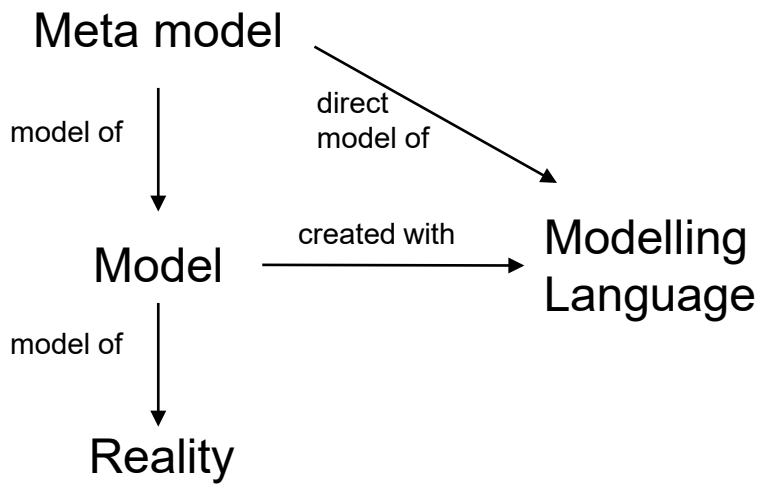


Modelling Language

Model



Meta-model



- A meta-model defines ...
  - … Concepts that can be used to create a model
  - … Attributes of concepts
  - … Rules to combine concepts
- The meta-model represents the general knowledge about the domain

Illustration: Modeling Language for Business Processes

Metamodel:

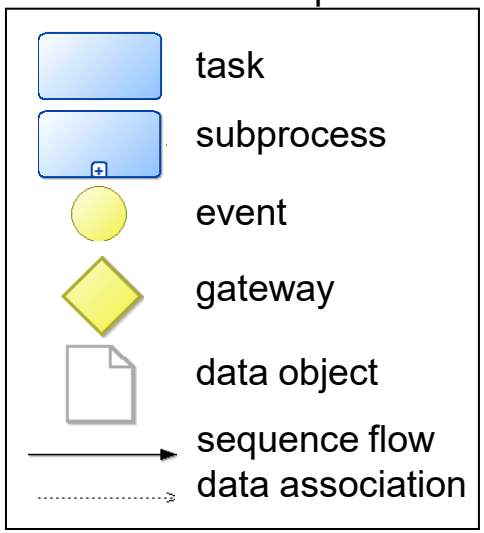
Concepts which can be used to create models.

Example: A process model consists of concepts for

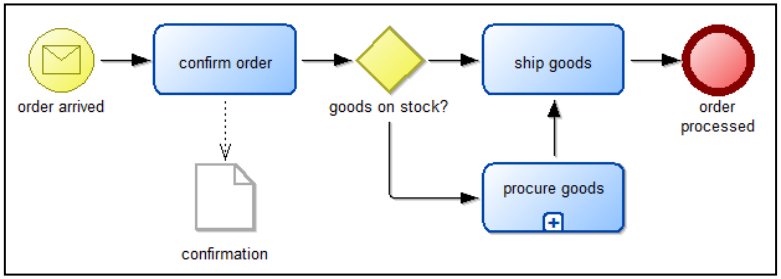
- Model elements: event, task, subprocess, gateway, data object
- Relationships: sequence flow, data association.

Modelling Language:

Notation/appearance of meta-model concept



Model:

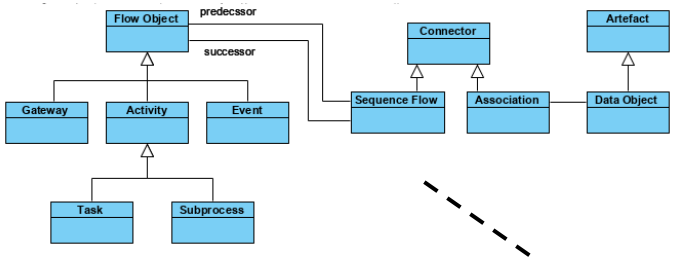


A model contains instances of the concepts defined in the meta-model. The object „confirm order“ represents a real entity; it is an instance of the concept «task"

Metamodels can be defined as Class Diagrams

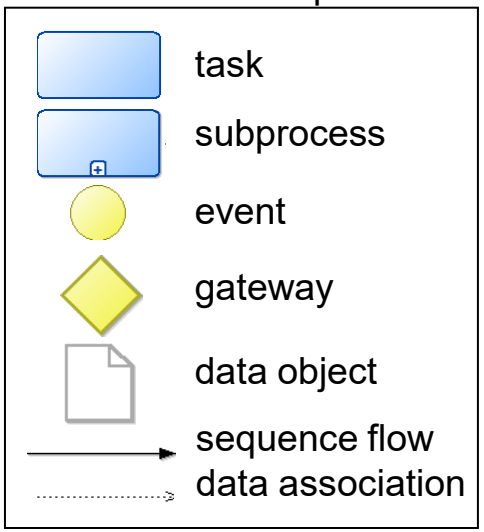
Metamodel:

Concepts which can be used to create models.

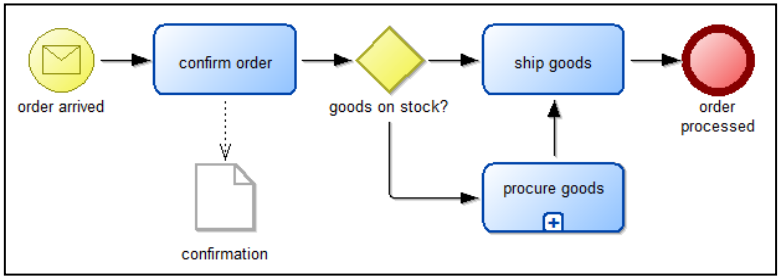


Modelling Language:

Notation/appearance of meta-model concept



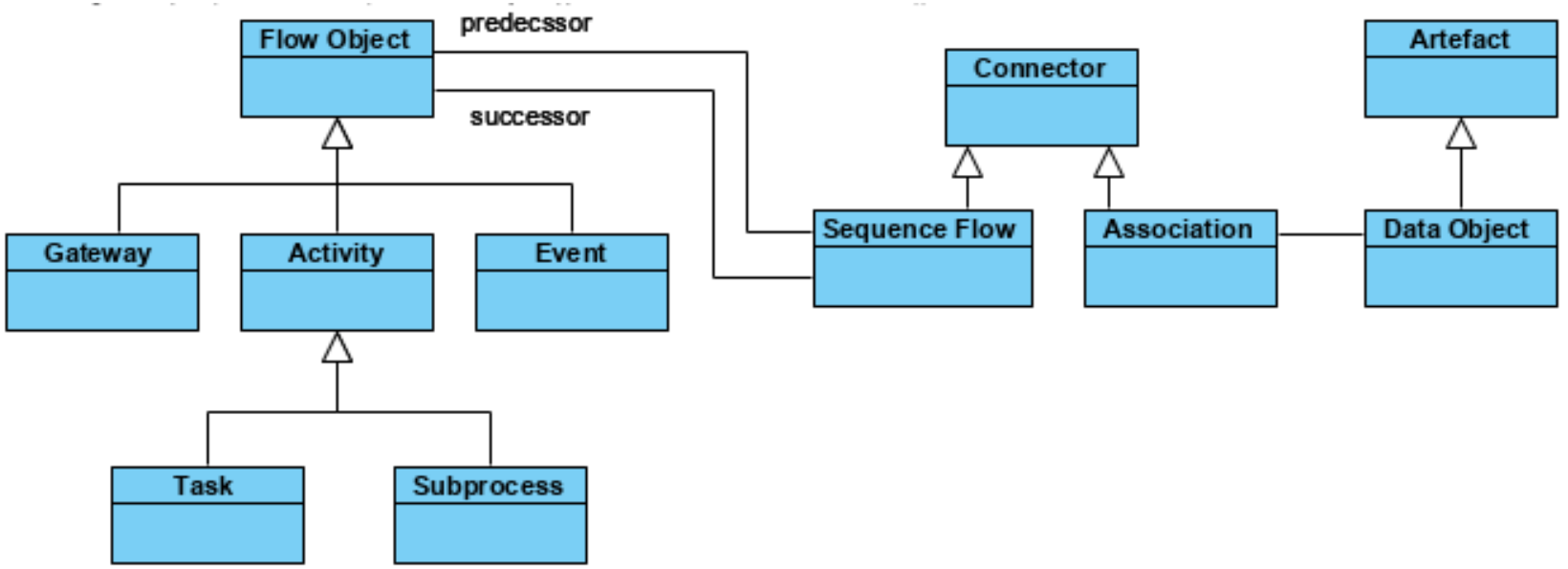
Model:



A model contains instances of the concepts defined in the meta-model. The object „confirm order“ represents a real entity; it is an instance of the concept «task"

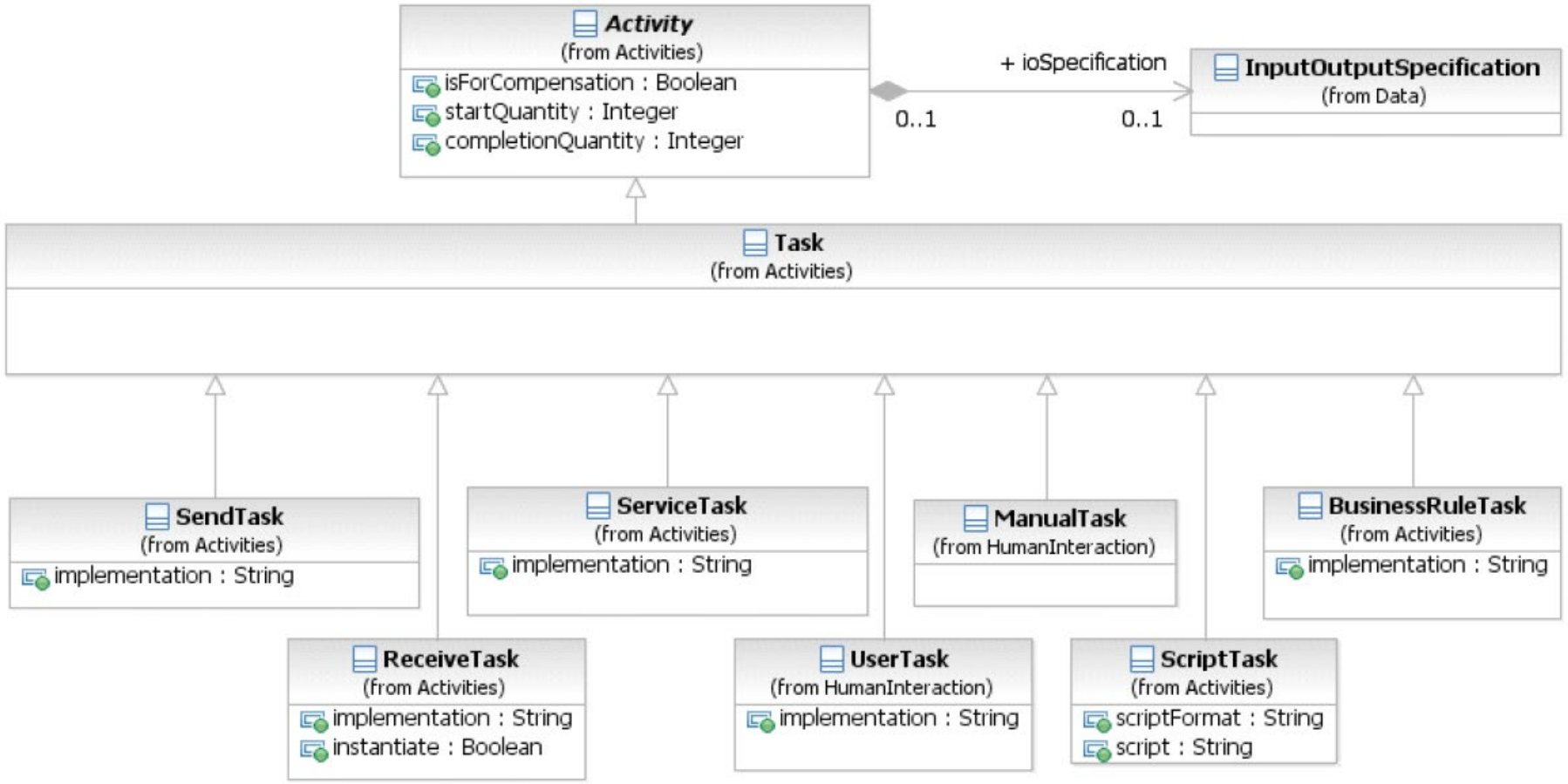
Metamodels can be defined as Class Diagrams

A Metamodeling language one can described meta models
Metamodels can be represented as class diagrams



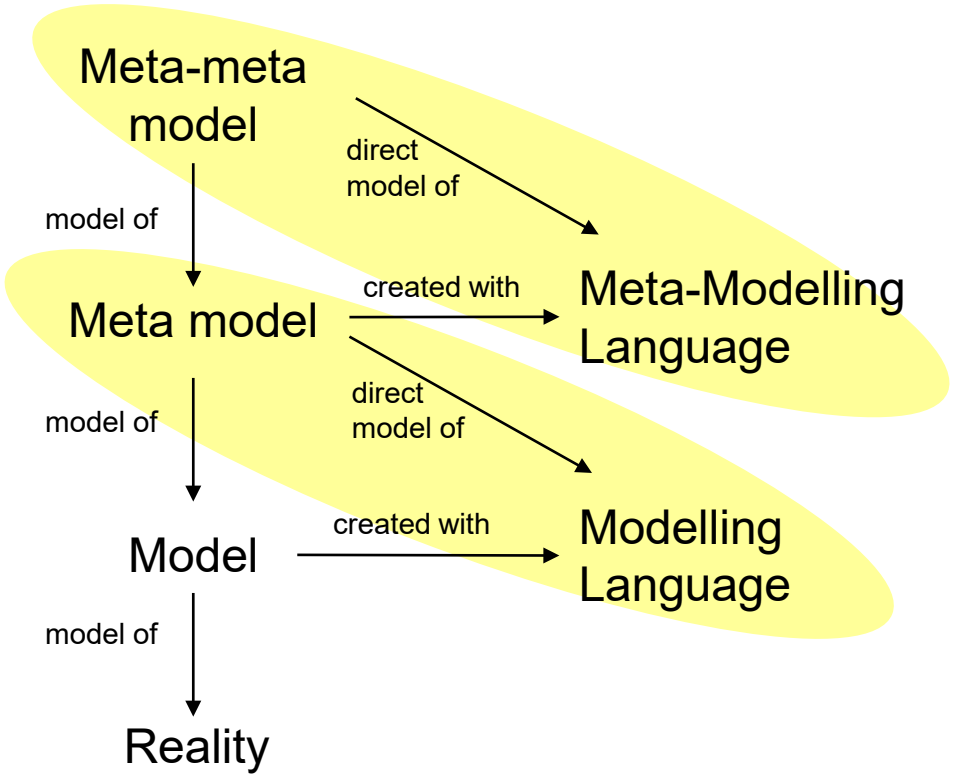
(UML Class diagrams where originally designed for modelling in object-oriented programming. This is why they contain operations and other features, which are not relevant for most modelling languages)

Subset of the BPMN Metamodel as UML Class Diagram



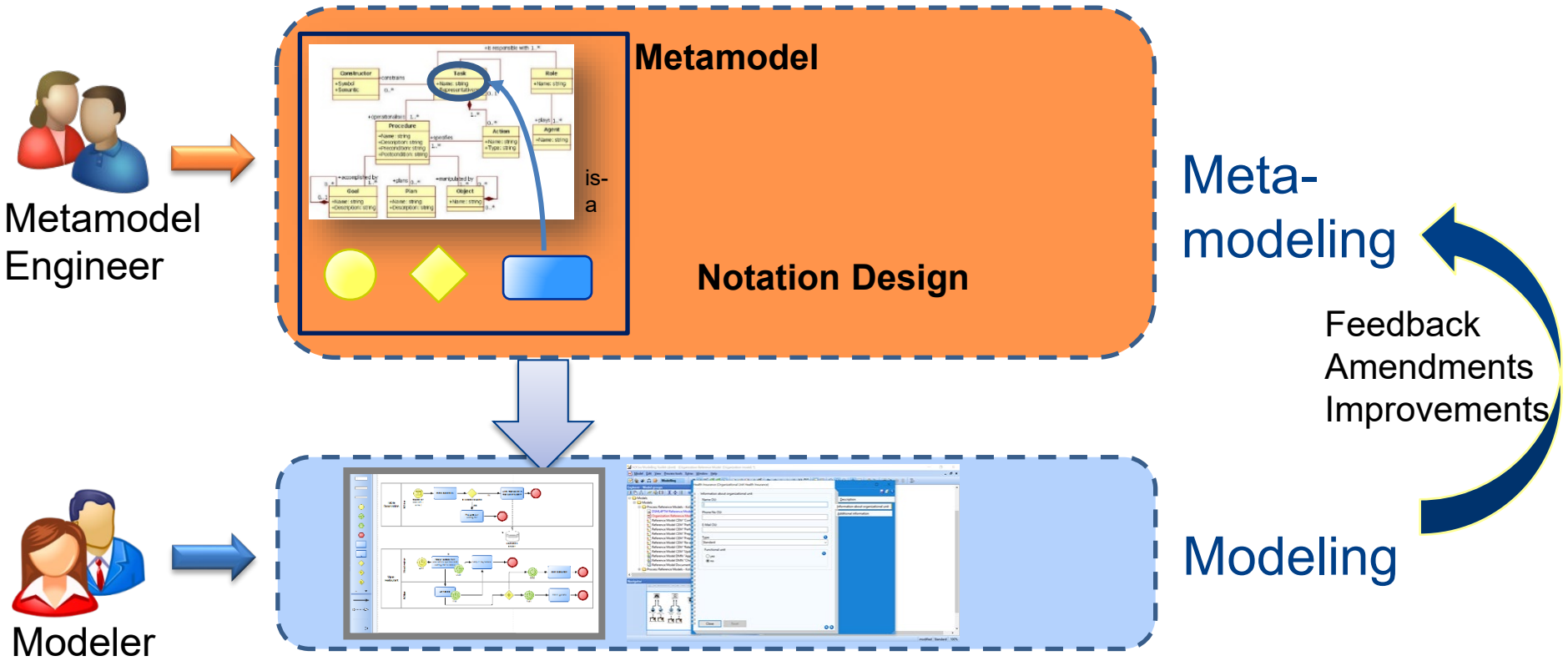
Source: BPMN 2.0 specification

Meta-meta model



- The meta model must again be described in some language, which is specified in a meta-meta model
- A **meta-meta model** defines the concepts for describing a meta model
- Graphical models usually have to kinds of concepts
  - Modeling elements
  - Relationships
- Examples for meta-modeling languages are
  - class diagrams.
  - Knowledge graphs
- Note: Meta-modeling languages are general-purpose modeling languages

Modeling and Metamodeling



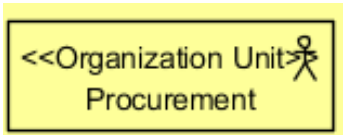
What do we do if there is no Domain-specific Modelling Language

- If there is no appropriate domain-specific modelling language for a domain of interest, we can
  - Metamodelling
    - Customization of an existing modeling language
    - Define a new domain-specific modelling language
  - Use a general-purpose modeling language

Customization of a Modeling Language

Customization in Archimate

- ArchiMate can be customized for specific usage by
  - Specialization of elements and relationships
  - Adding specific attributes, e.g. for cost calculations
- In Visual Paradigm the specialization can be done with stereotypes
  - Example: Specialization of Actor



Examples of Specializations

Specializsation can be made for elements and relations on all layers

- A **Business Actor** could be
  - Individual
  - Organization Unit
  - Company
- **Product** could be
  - Physical Product
  - Digital Product
- A **Data Object** coud mean
  - Document
  - Structured Data
- **Application Interface** coud be
  - Application-to-Application Interface
  - User Interface
- **Network** could be
  - WiFi Network
  - Wide Area Network
- **Equipment** could be
  - Machine
  - Vehicle

Using a General-Purpose Modeling Language

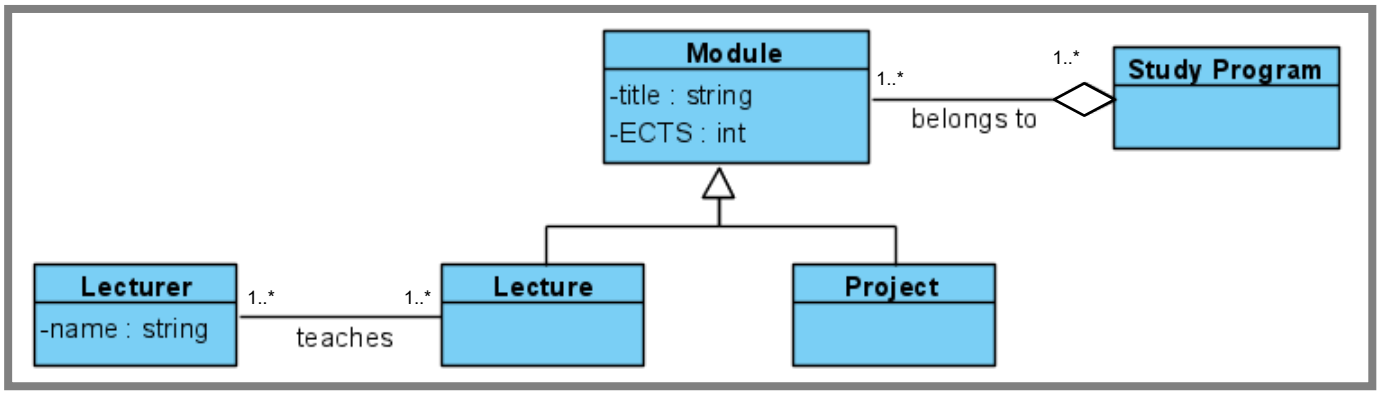
General-Purpose Modeling Languages

- 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
  - Class Diagrams
    - Artificial Intelligence: Ontologies, Knowledge Graphs
    - Data Modeling: Entity Relationship Diagrams
    - Object-Oriented Programming: UML Class Diagrams

Modeling with a General-purpose Modeling Language

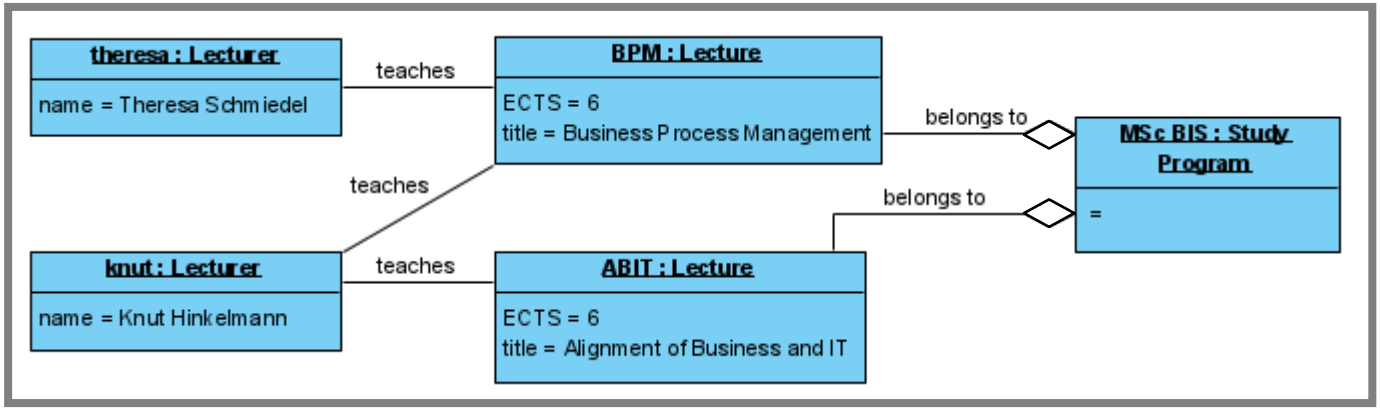
- UML Class Diagrams are general-purpose modeling languages
- A class diagram contains **Classes** with **Attributes** and **Associations**

Classes
(=metamodel)



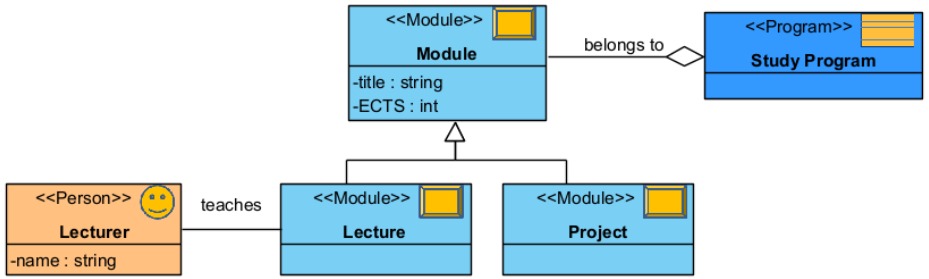
- A model consists of objects which are instances of these classes

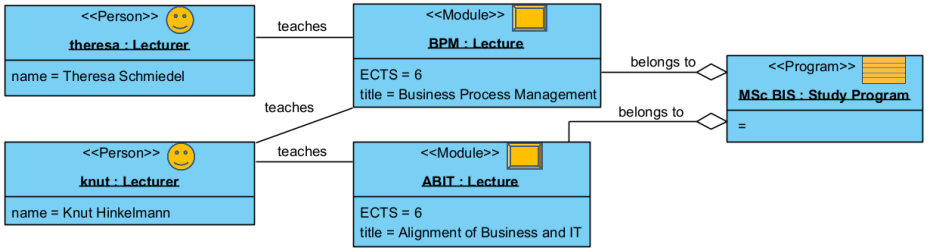
Objects
(= model)



Customization of a General-purpose Modeling Language

- 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 also to specialize UML class diagrams.
- We can define a new stereotype for a class and change color or add an icon
- Example: stereotypes for modules and lecturers



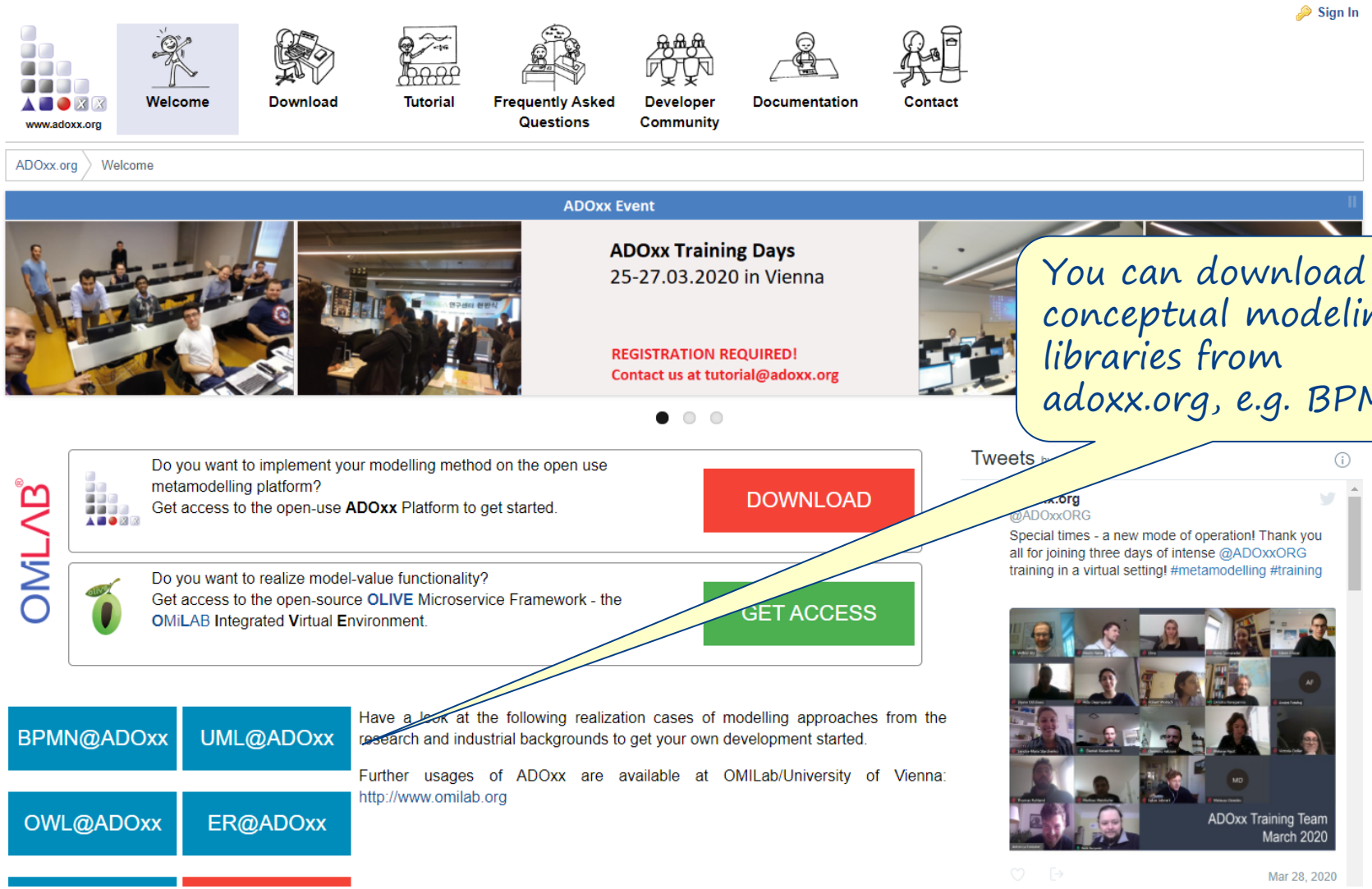


Strengths and Weaknesses of General-Purpose Modeling Languages

- 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
  - Restricted reusability
    - Different applications use different concepts

Creating a new Modeling Language: Metamodelling with ADOxx

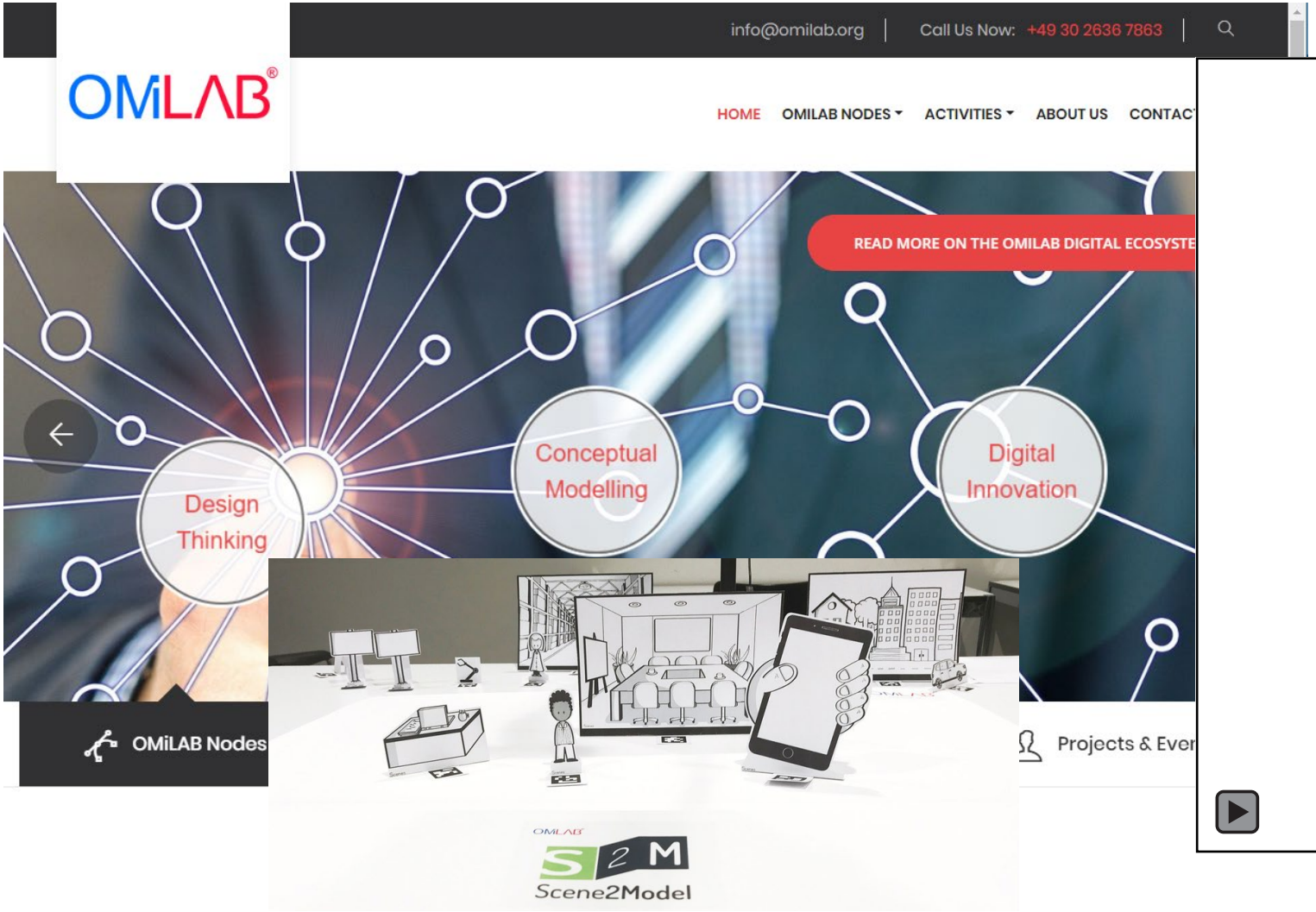
adoxx.org – Download, Tutorials, Community



You can download conceptual modeling libraries from adoxx.org, e.g. BPMN,

OMiLAB – A Conceptual Modelling Community

ADOxx is the basis for OMiLAB

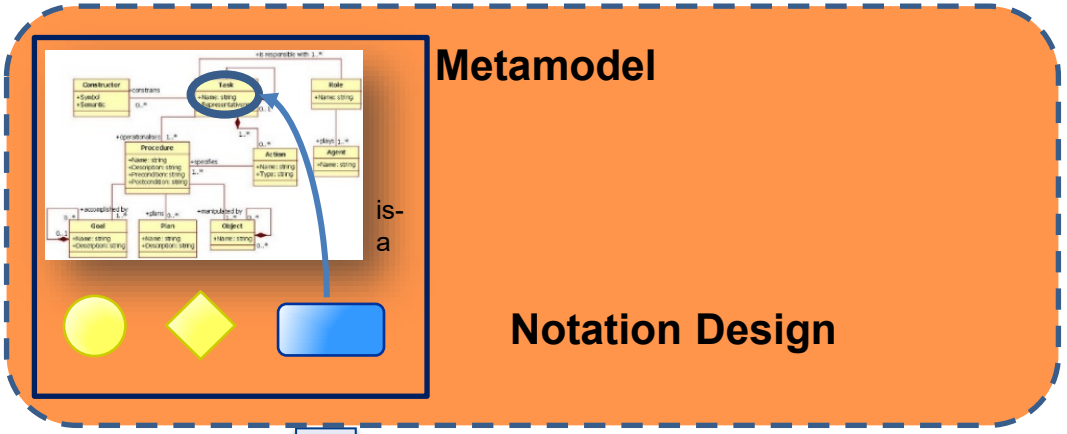


The ADOxx Environment

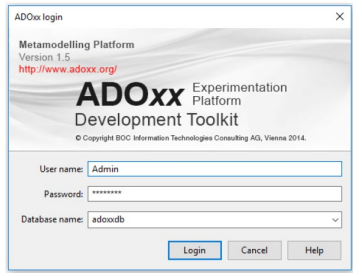
- ADOxx consists of …
  - ADOxx Development Toolkit
    - Defining Modelling languages – Library Management
    - Administration of users, models, components
  - ADOxx Modelling Toolkit
    - Creating models

Modeling and Metamodeling

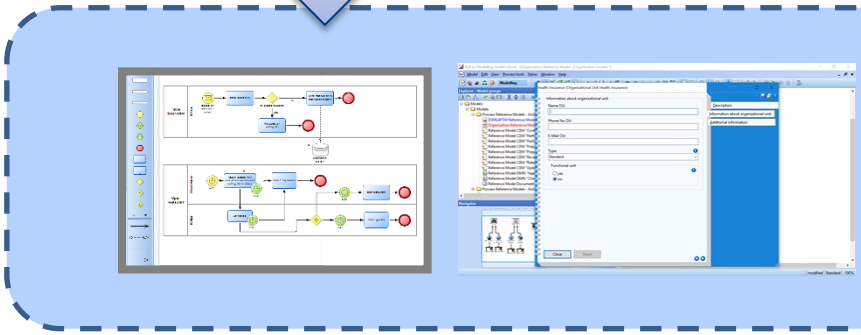
Metamodel Engineer



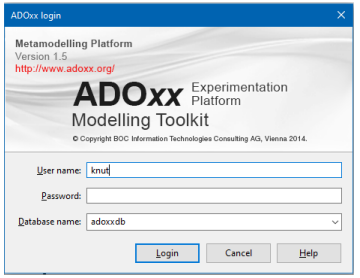
Meta-modeling



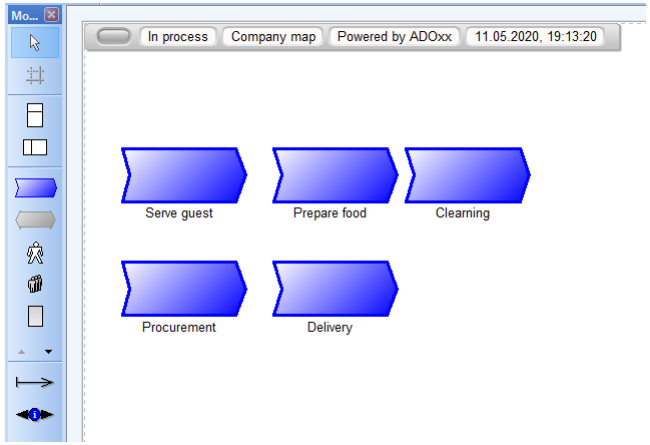
Modeler

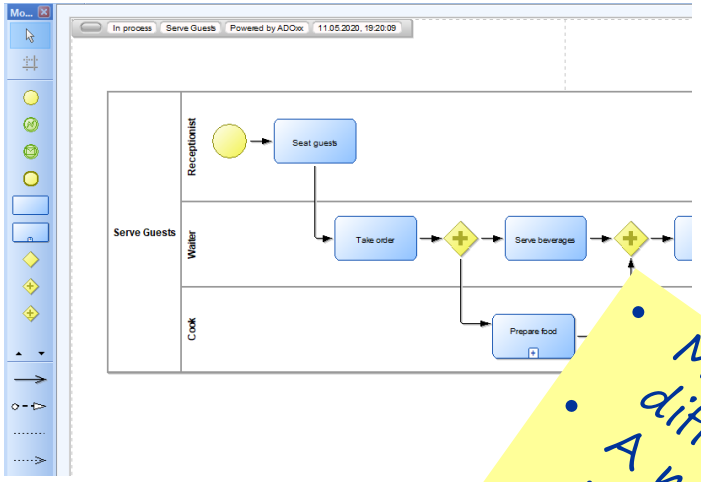


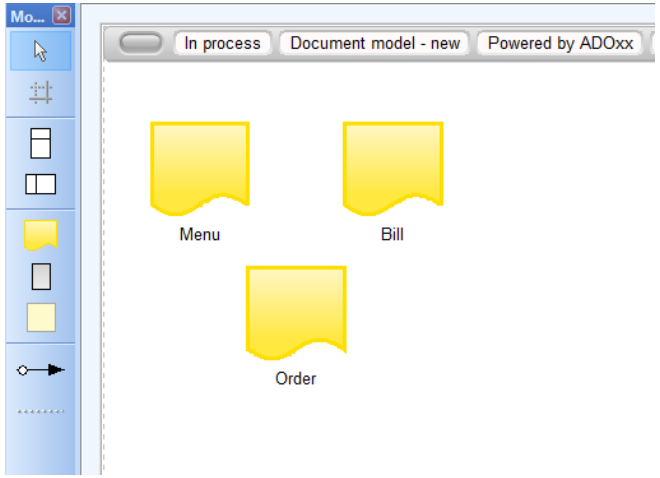
Modeling

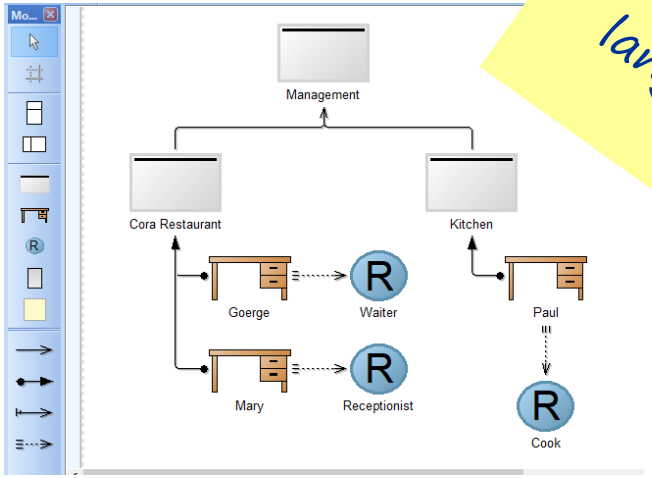


A Modleling Environment can consist of several Model Types





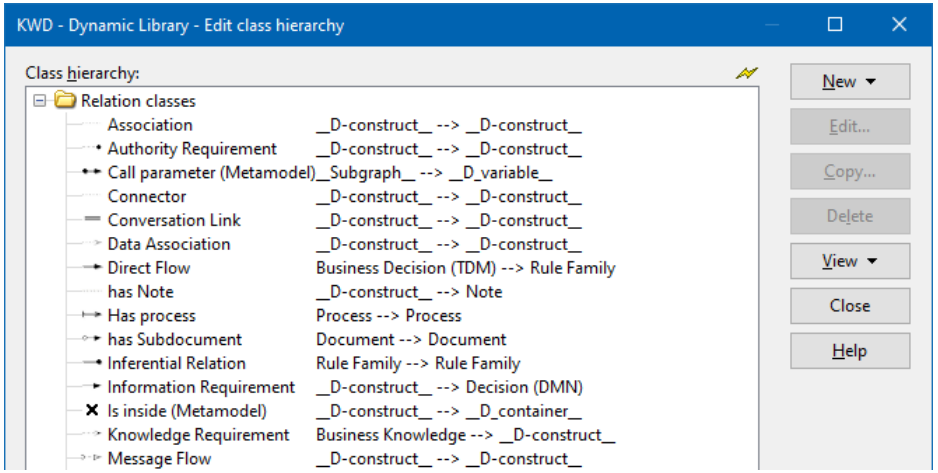


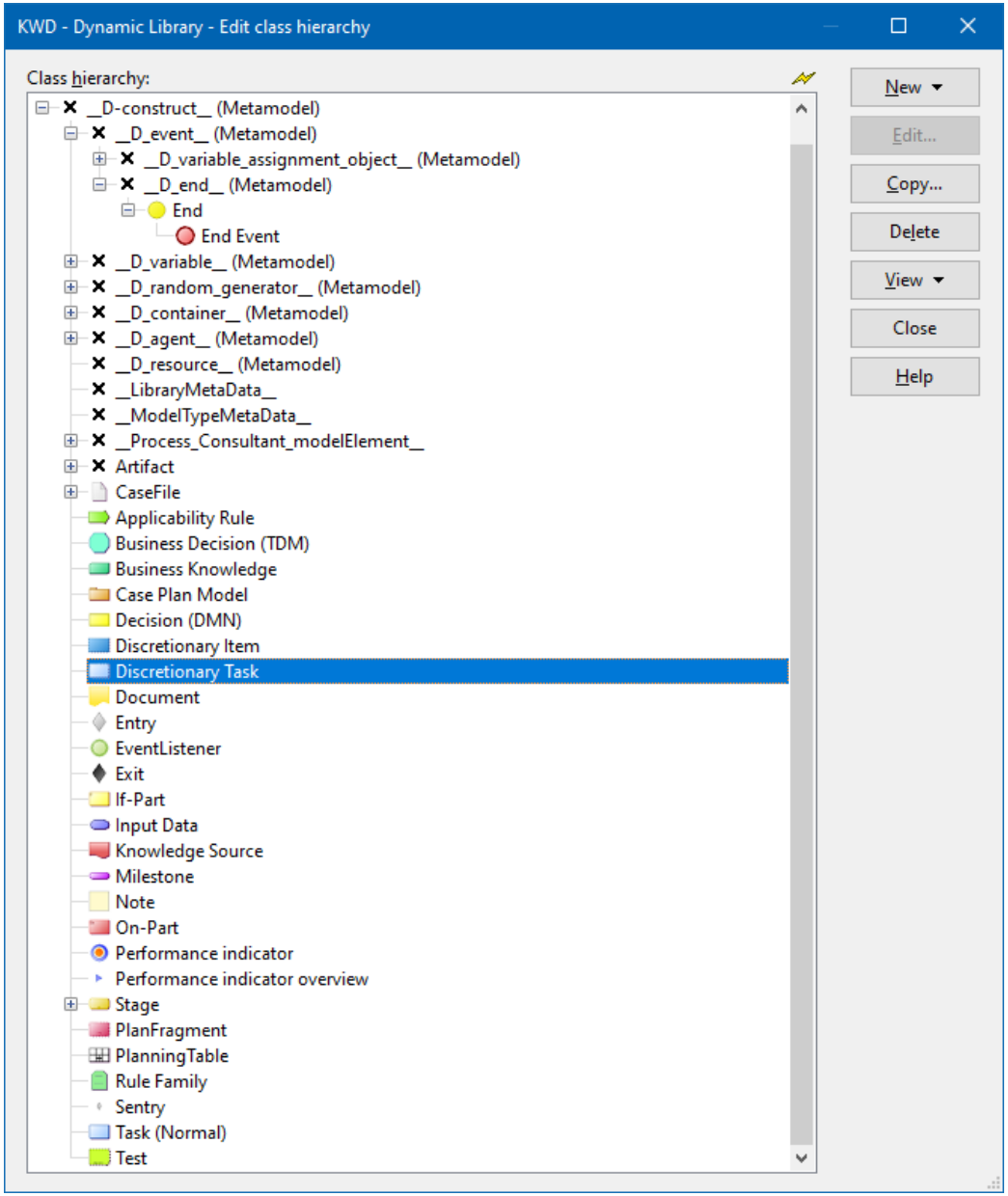


- Model Types represent different views
- A model type can be regarded as a modeling language of its own.

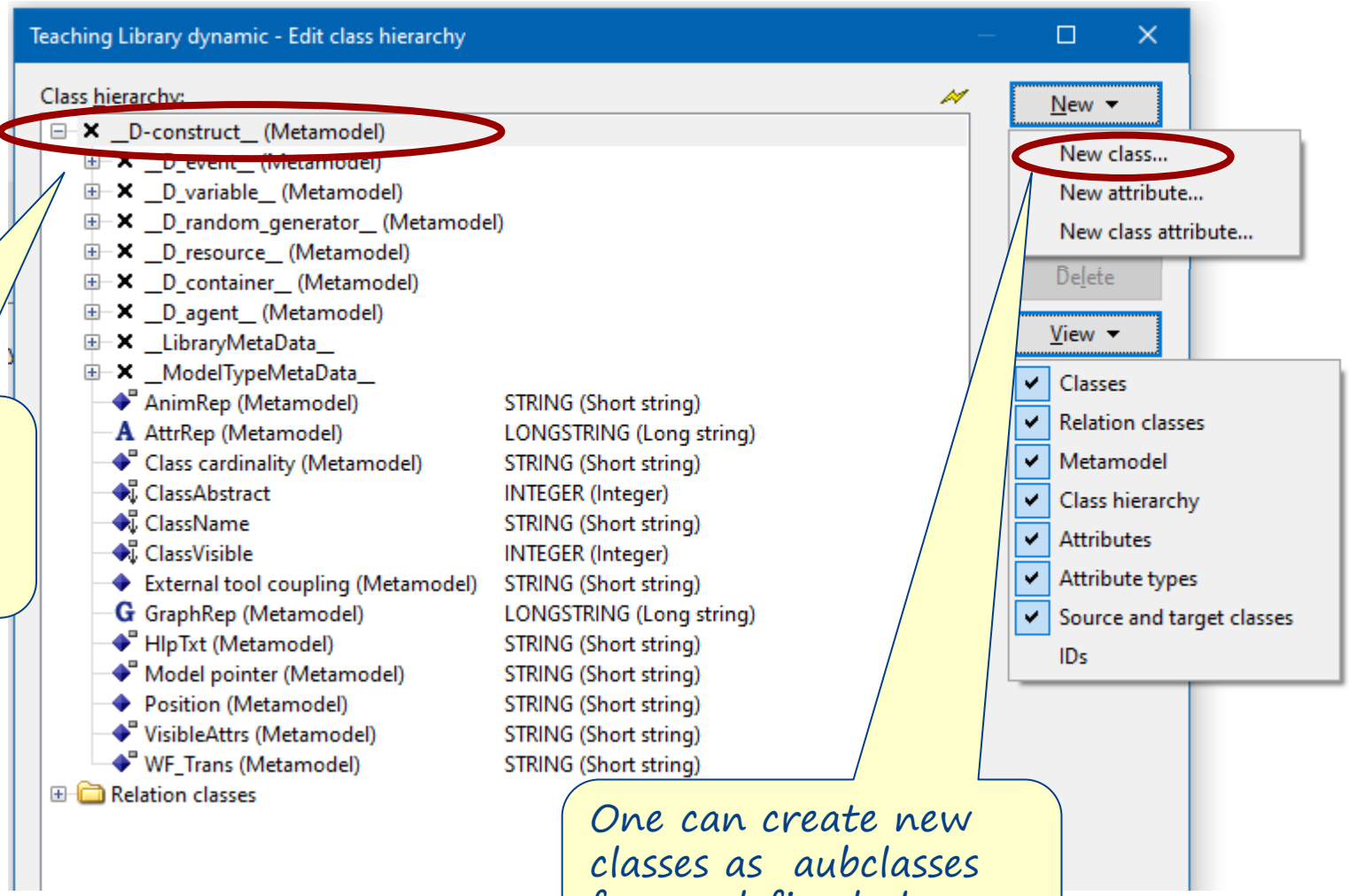
Class Hierarchies

- ADOxx distinguishes
  - Classes
  - Relation classes





Creating new Classes

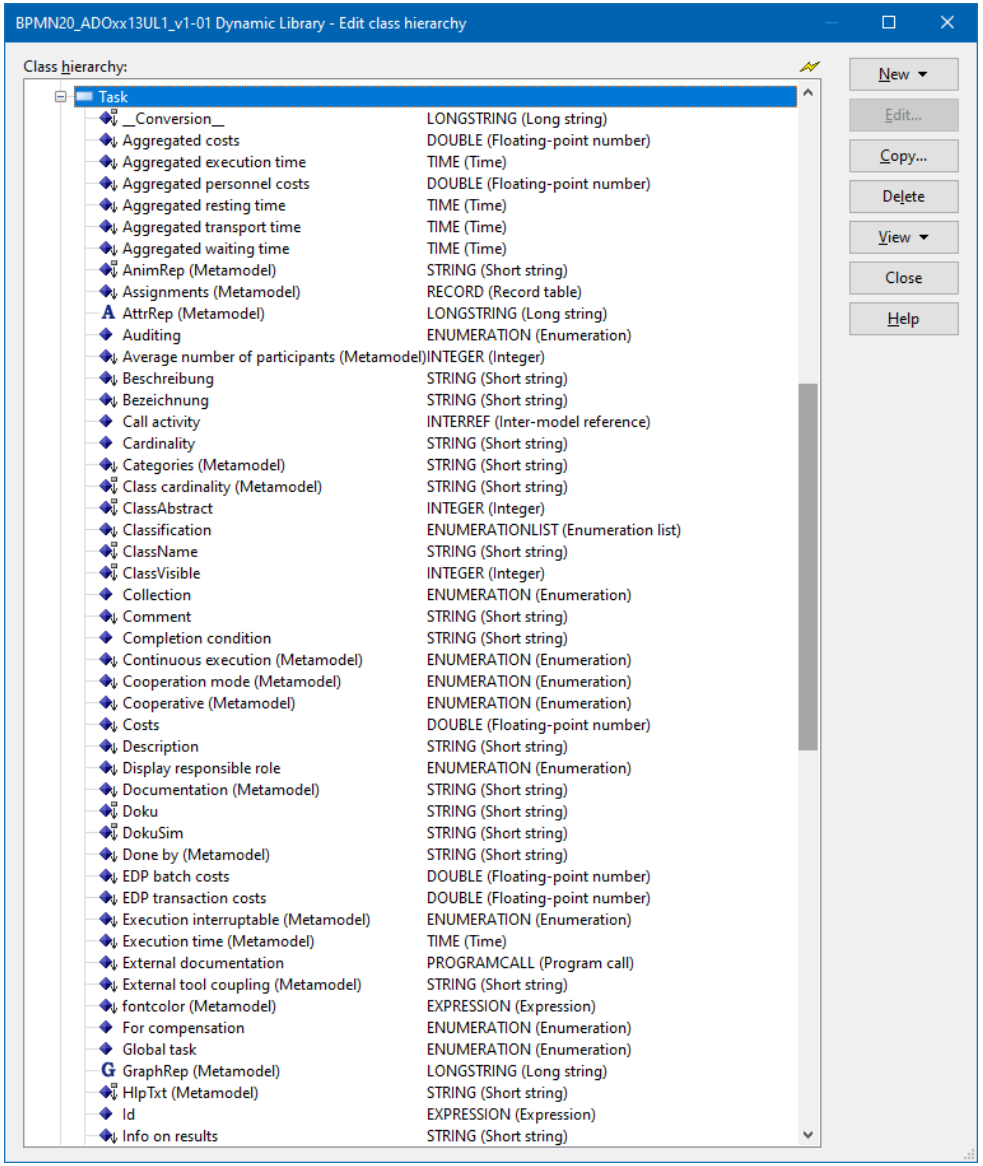


There are predefined abstract classes which have specific functionaliy

One can create new classes as aubclasses for predefined classes and relations classes

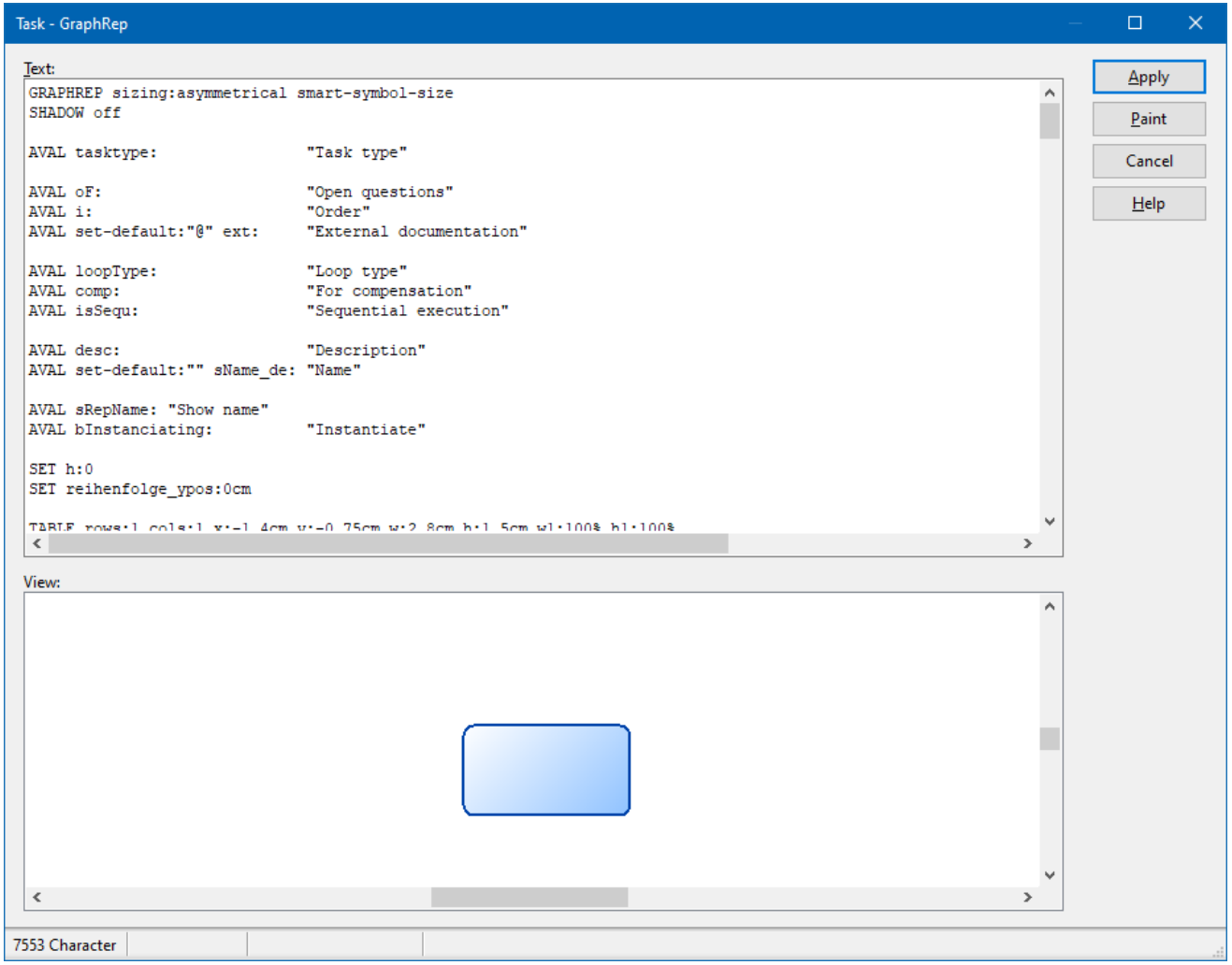
Attributes

- Classes and Relations have Attributes
  - Properties of Elements
  - Graphical Representation
  - References



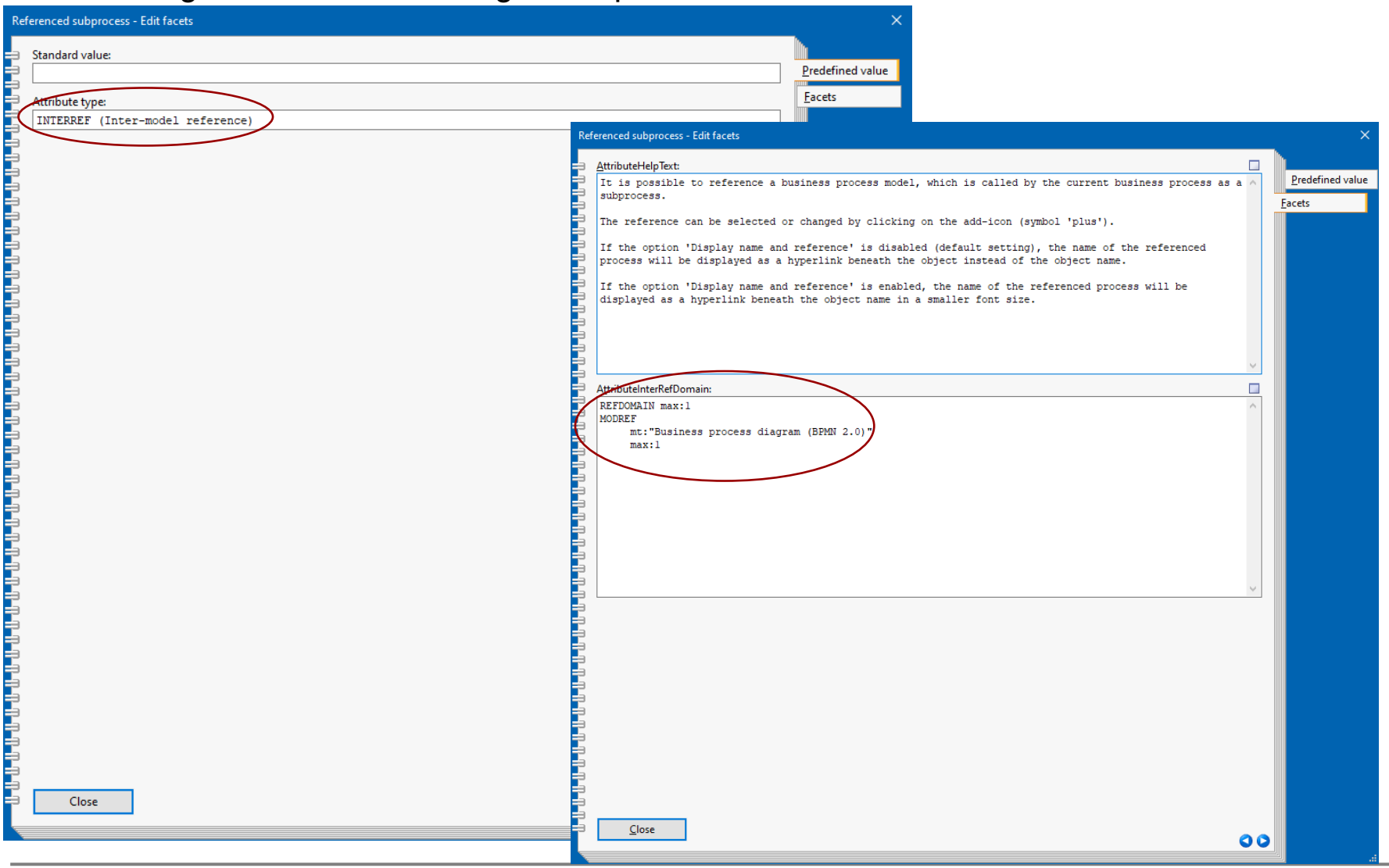
Special Attribute GraphRep

GraphRep: A script language for the graphical representation

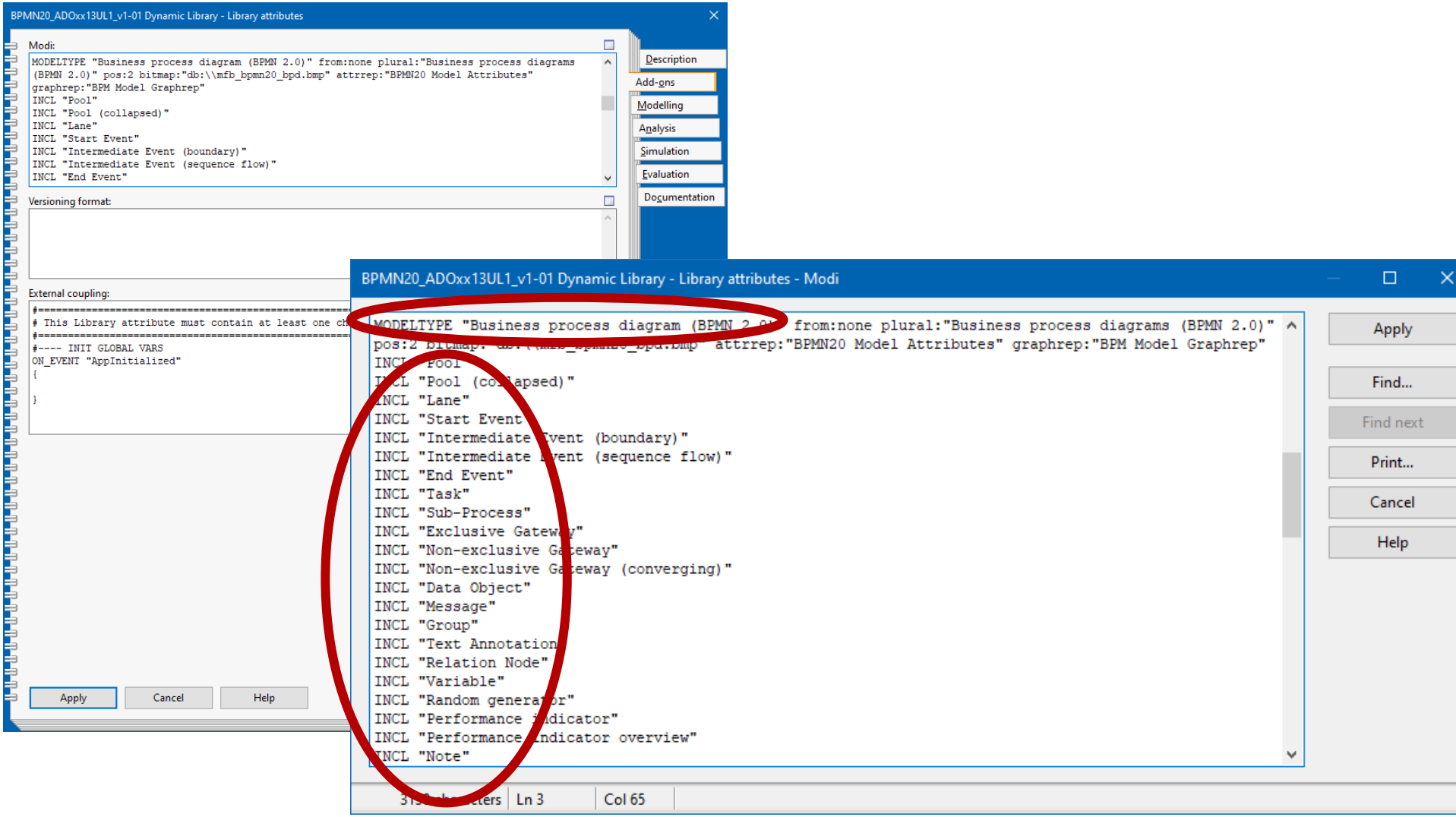


References

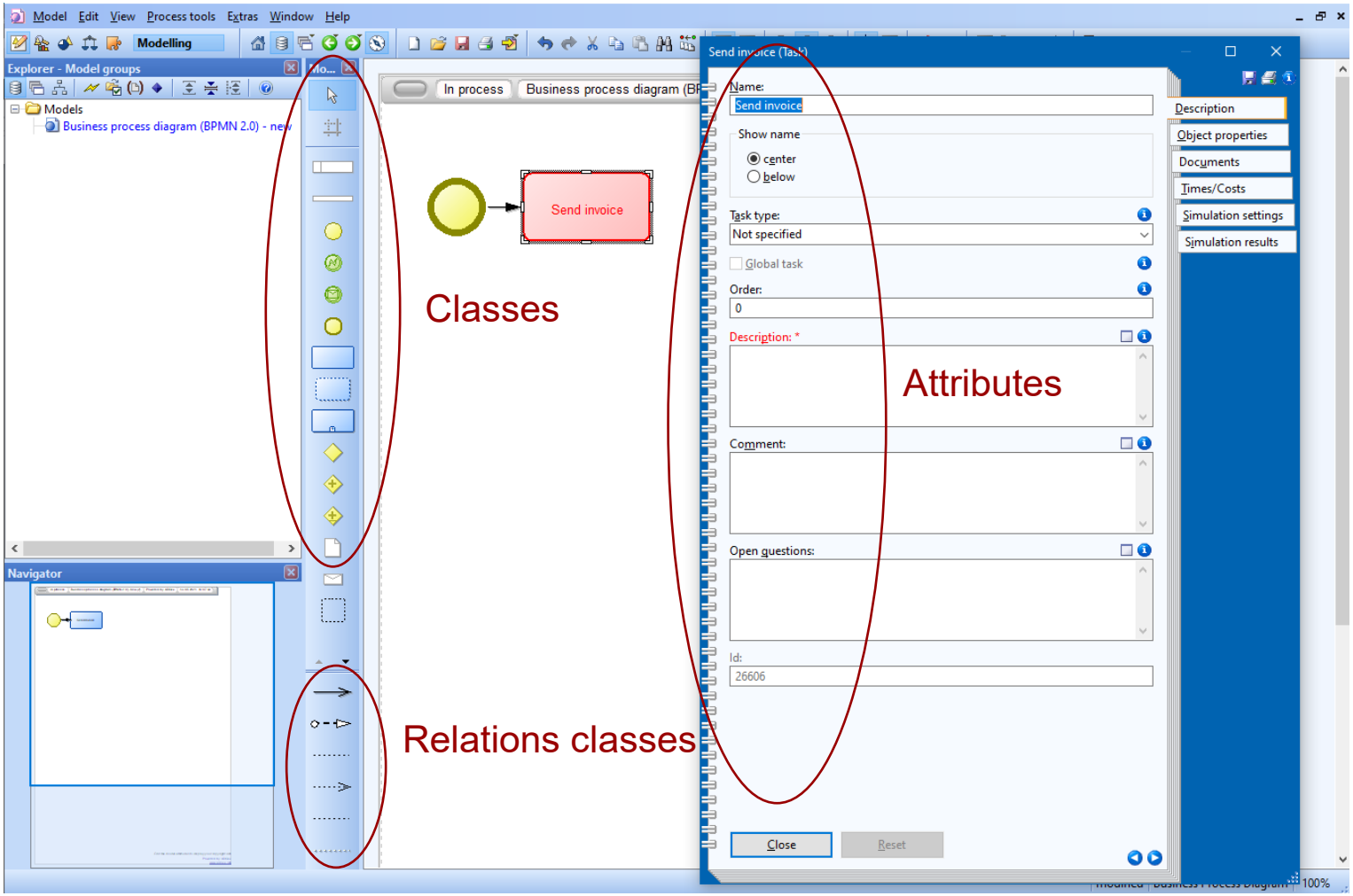
Referencing a another model, e.g. a subprocess



Classes and Relationss are assigned to Model Types

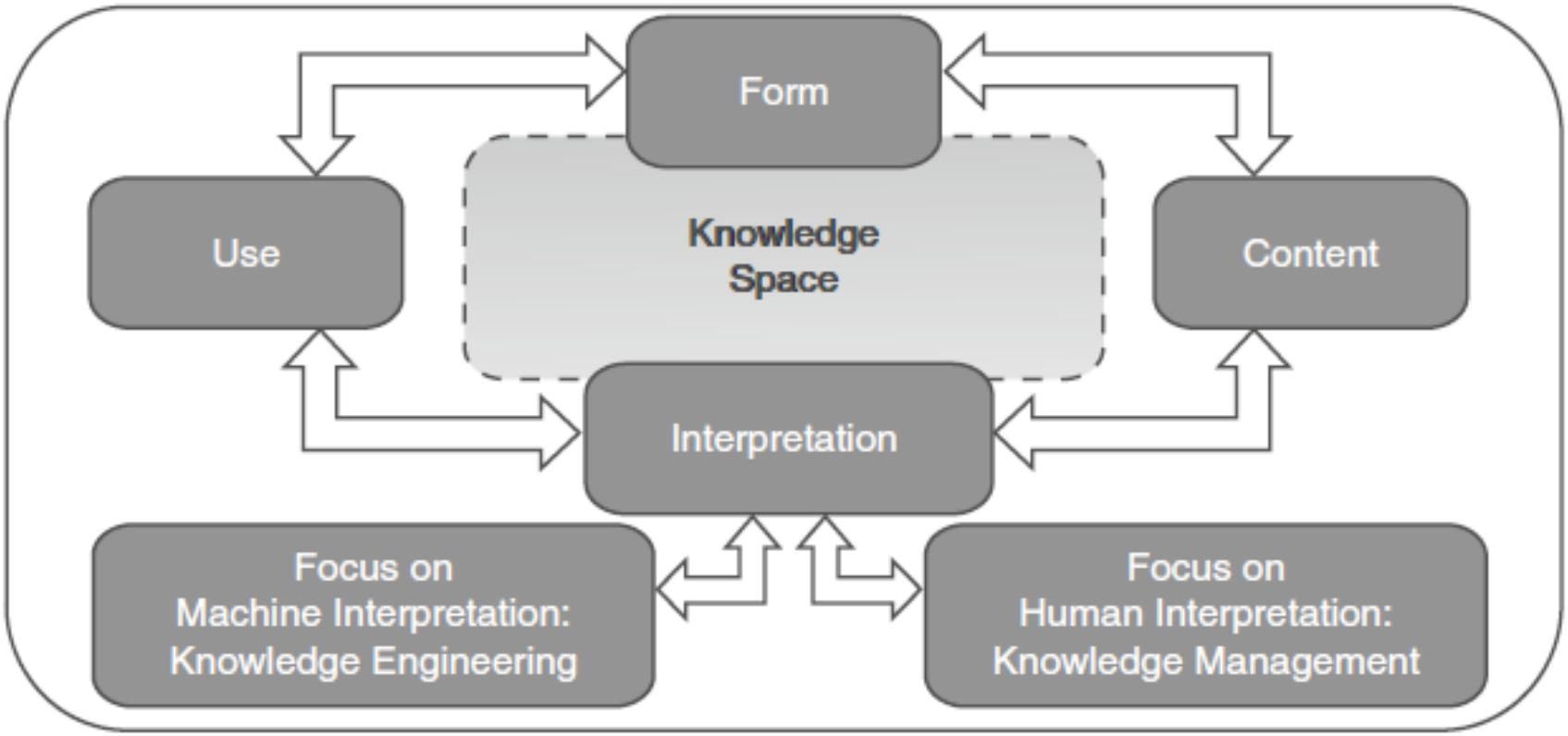


Appearance of Classes and Attributes in the Modelling Toolkit



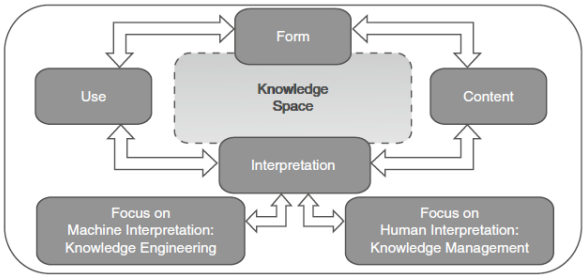
Knowledge in Models

Dimensions of a Knowledge Space



Karagiannis, D., & Woitsch, R. (2010). Knowledge Engineering in Business Process Management. In *Handbook on Business Process Management 2* (pp. 463–485). Springer.

Dimensions of the Knowledge Space



Use:
- process optimization requires knowledge about time and costs
- selection of a cloud service require knowledge about data and functionality

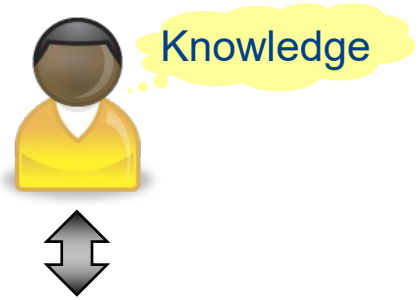
Form: modeling language

Content: Instantiation of concepts

- ***Use***: Stakeholders and their concerns determine the relevant subset of the knowledge
- ***Form***: Syntax and semantic of *predefined concepts*.
- ***Content***: *Instantiation* of predefined concepts for a specific *application* (represented in the labels)
- ***Interpretation***: Giving meaning to a model:
  - Graphical models are cognitively adequate for human
  - Machines need more formal representation

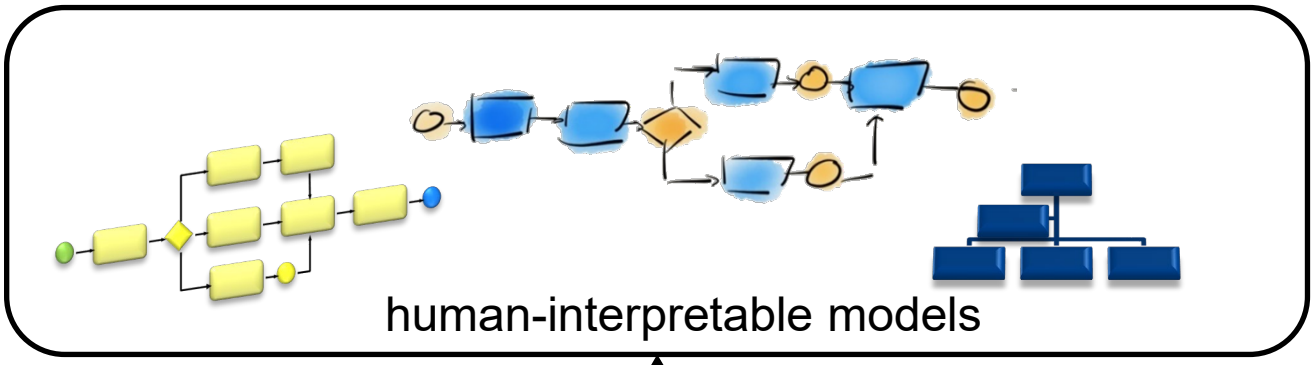
Graphical Models are appropriate for Humans

Communication/
Analysis/
Decision Making



Examples:
Visio
Powerpoint

Models



Reality

Making the Knowledge in Models explicit

- Humans «know» the meaning of the modeling objects.
  - ♦ Elements of the model language
  - ♦ Labels represent domain knowledge
- Examples:

♦Model element: Application Compontent
♦Domain: «ERP System» is business software

♦Model element: Task
♦Domain: «Cook pasta» is about preparing food

- The objective is to represent the knowledge so that it can be interpreted by a system for decision making and problem solving

Glossary

- A glossary is a collection of terms and their definitions.
- Example: Glossary in Visual Paradigm
  - Defining the terms used in the labels of models or in Business Rules

Cheapflight Glossary -> Glossary Grid

search...

Name	Aliases	Labels	Description
Aircraft	Airplane		A machine capable of flight.
Baggage	Luggage		Suitcases or other bags in which to pack personal belongings for travelling. It is distinguished between carry-on baggage and checked baggage
Customer	Client, Passenger		A person who bought or is going to buy a ticket
Ticket			Confirmation that the customer made a booking