

Ontology-based Enterprise Modeling for Human and Machine Interpretation

Prof. Knut Hinkelmann

knut.hinkelmann@fhnw.ch



Image by [Pexels](#) from [Pixabay](#)



Why Modeling



- If the object you want to create or change is simple, then you can do it directly.

- For **complex** systems that are likely to **change** over time, you need a **model**.

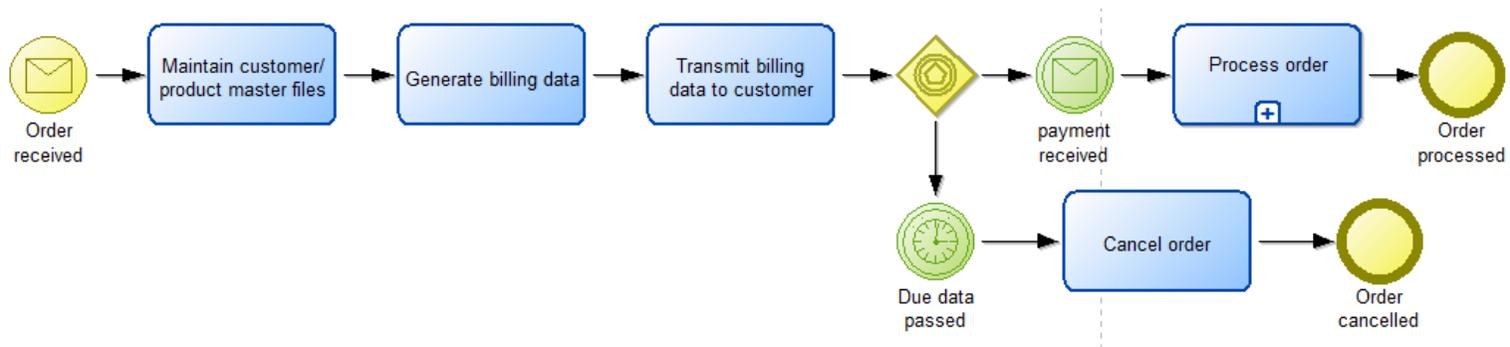


- Without explicit modeling there is a *high risk that the implementation is not what is intended*

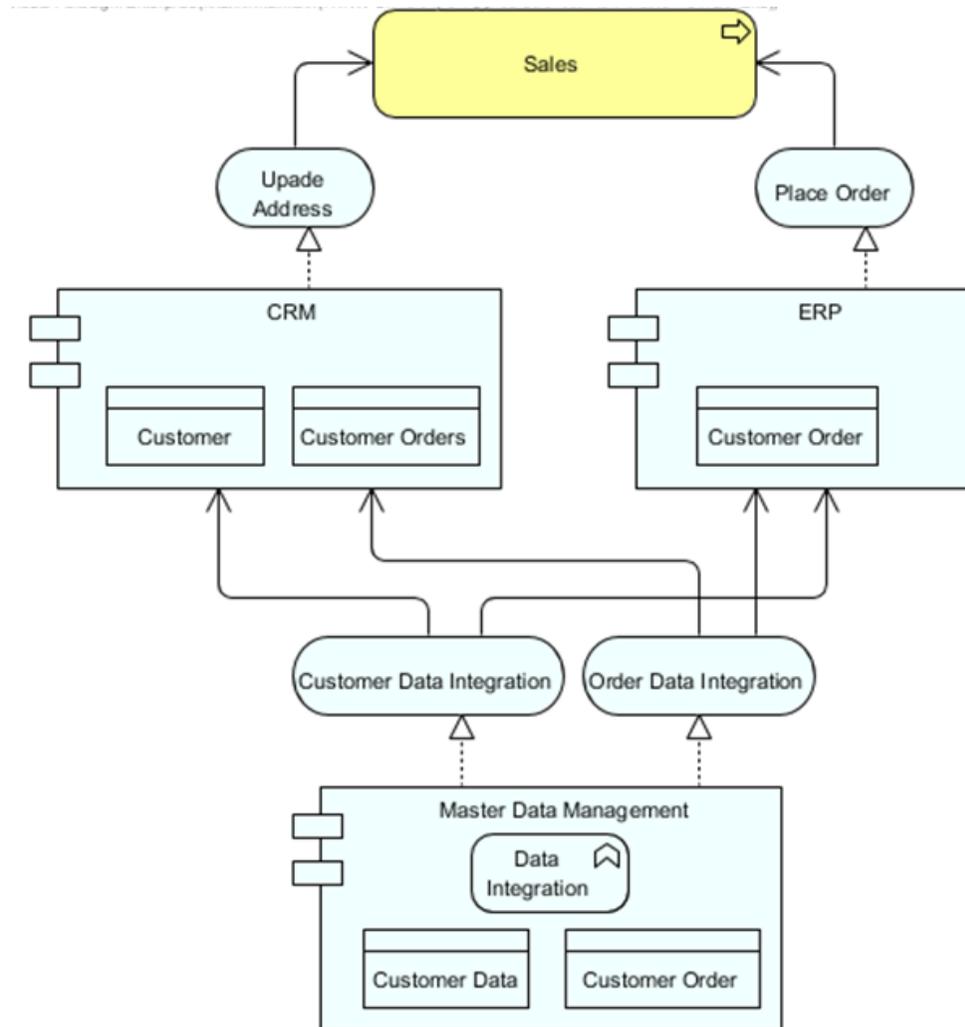
(John Zachmann, 2012)

Business Process Management

- Process Design
- Process Optimization
- Process Digitalization
- ...

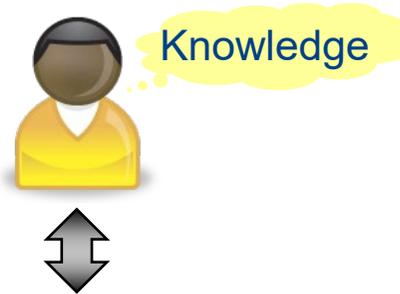


Enterprise Architecture

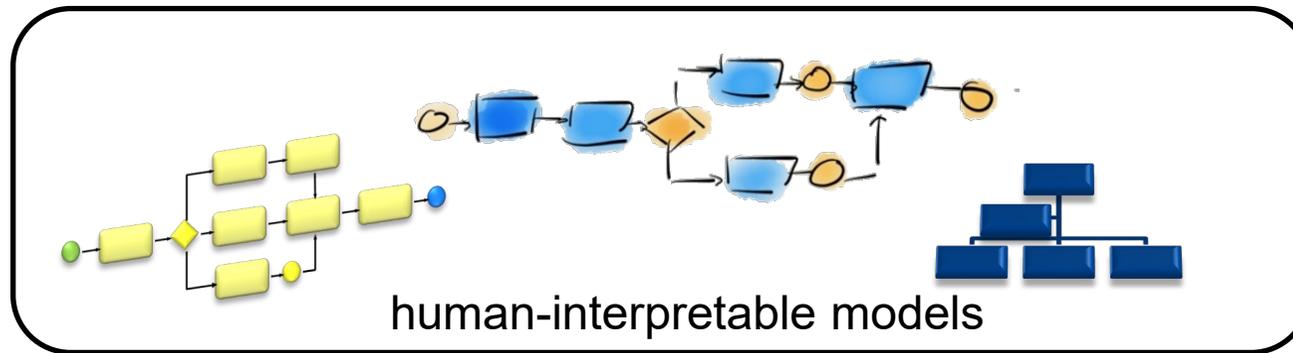


Graphical Models are appropriate for Humans

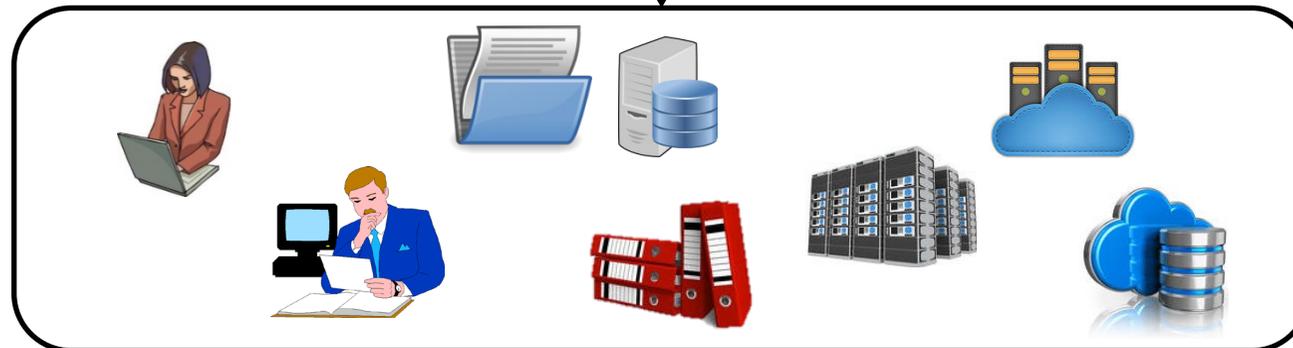
*Communication/
Analysis/
Decision Making*



Models



Reality

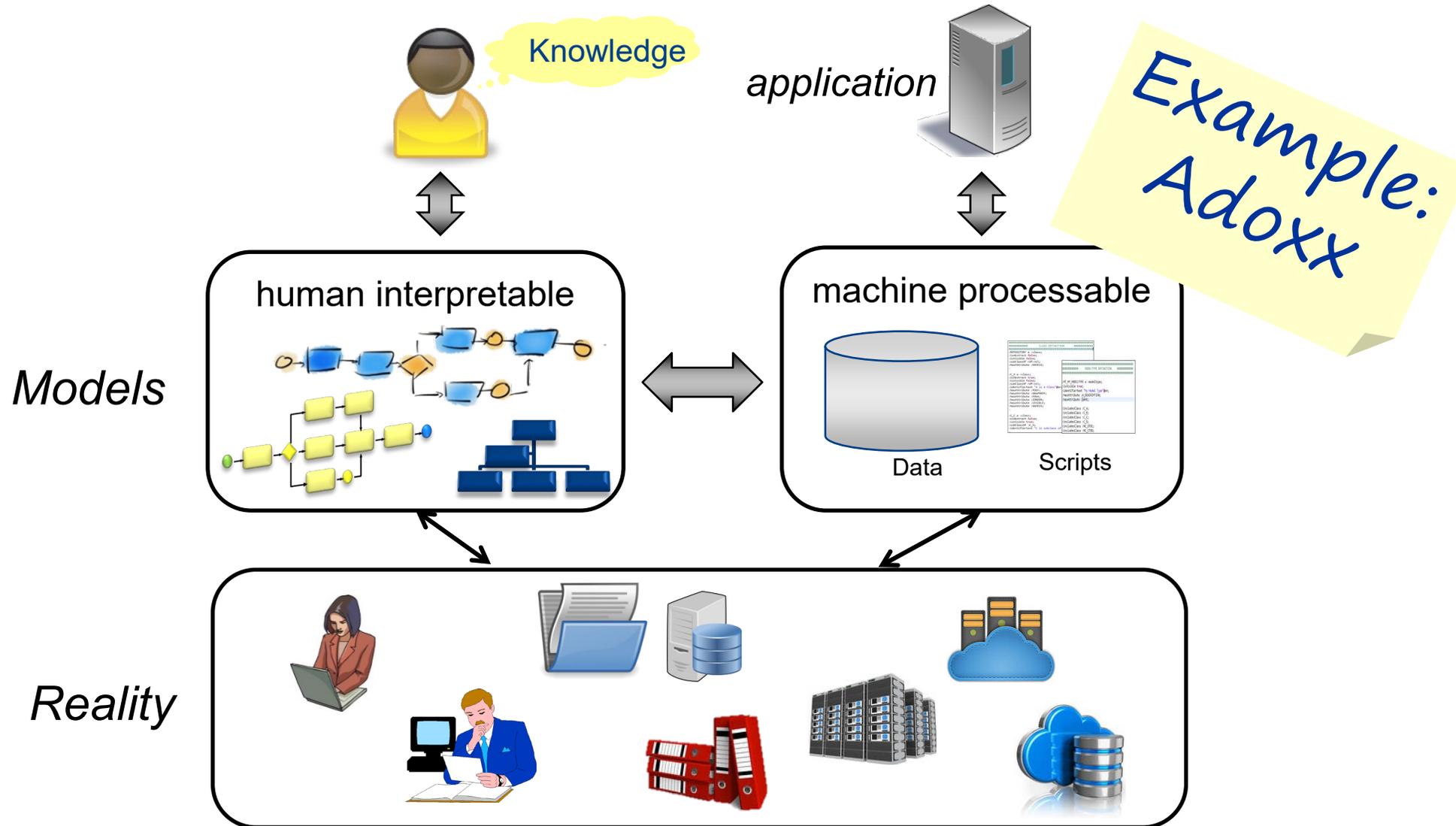


Models

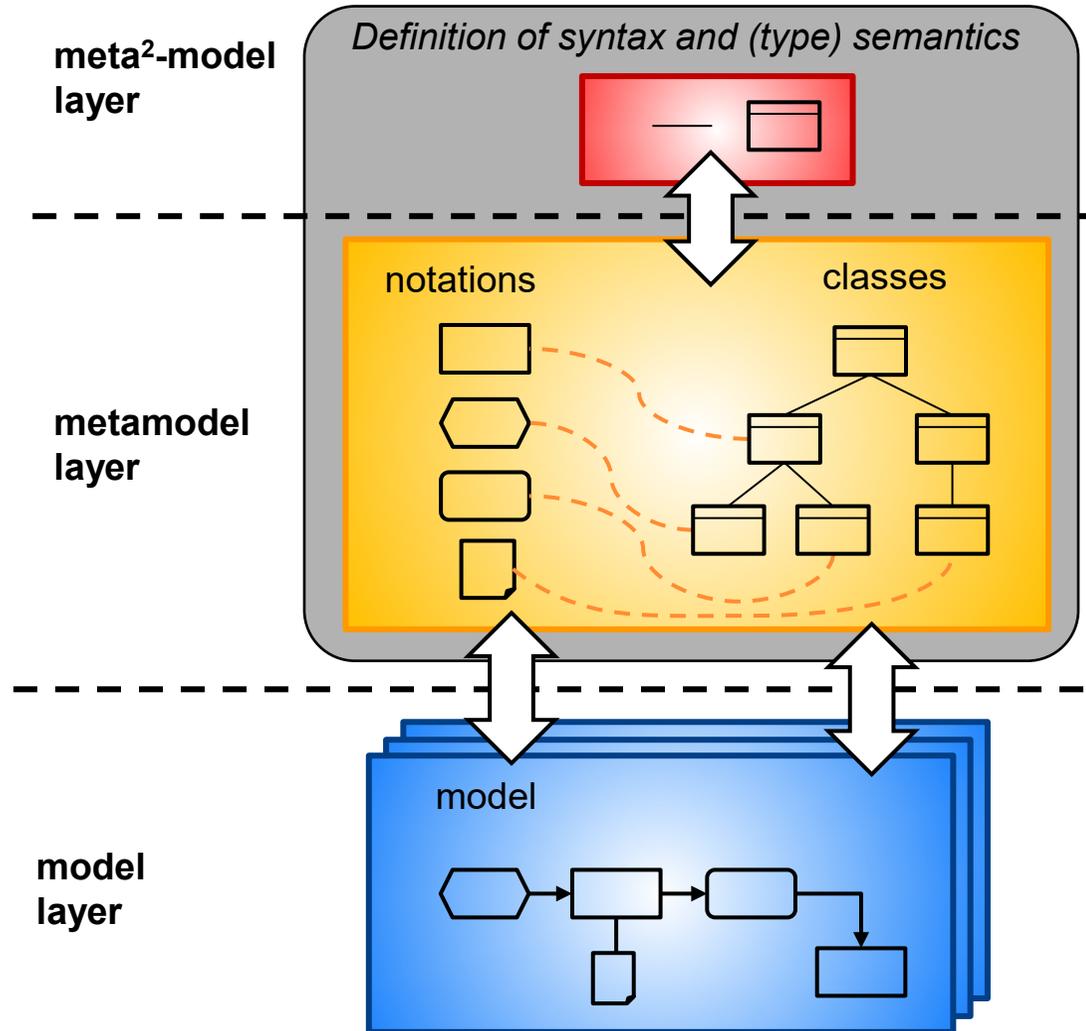
- Models are not mere pictures; rather, they
 - ◆ provide a precise, meaningful description that can be visualized in different ways for different stakeholders;
 - ◆ can also be used to analyze the impact of changes, cost, risk, security, compliance and other relevant KPIs.

Models should allow automated analysis,
decision making and digitalization

Graphical Models are Represented in a Database

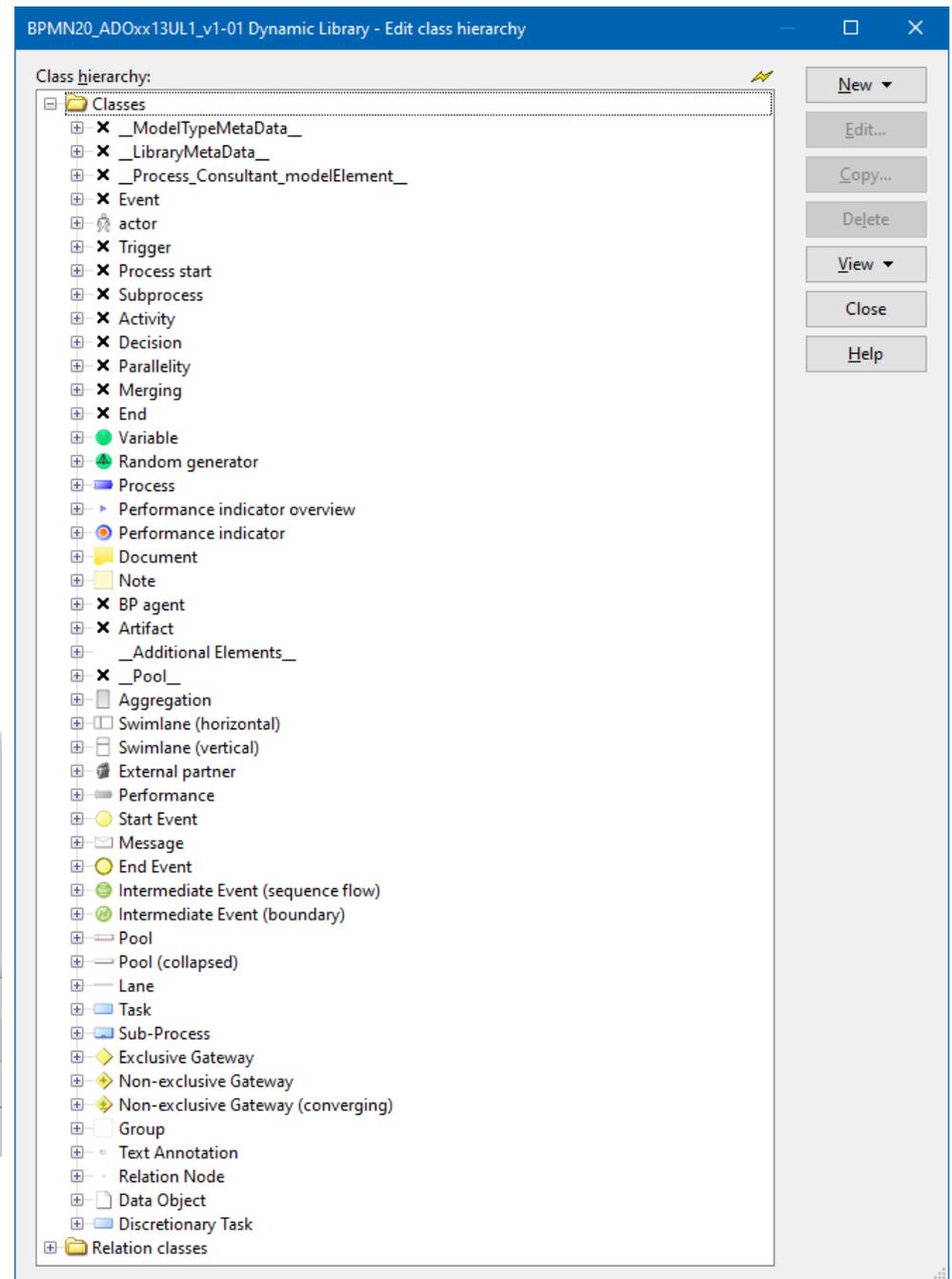
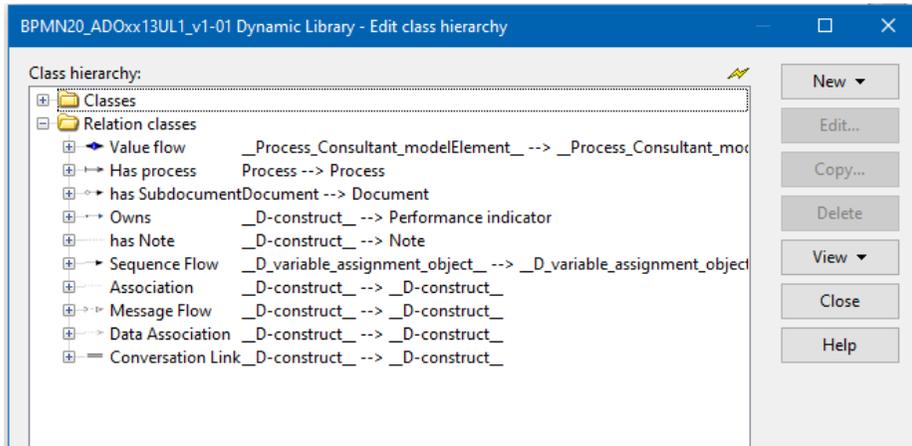


Modelling Environment



Class Hierarchies

- ADOxx distinguishes
 - ◆ Classes
 - ◆ Relation classes



Attributes

■ Kinds of Attributes

- ◆ Properties of Models
- ◆ Graphical Representation
- ◆ References

BPMN20_ADOxx13UL1_v1-01 Dynamic Library - Edit class hierarchy

Class hierarchy:

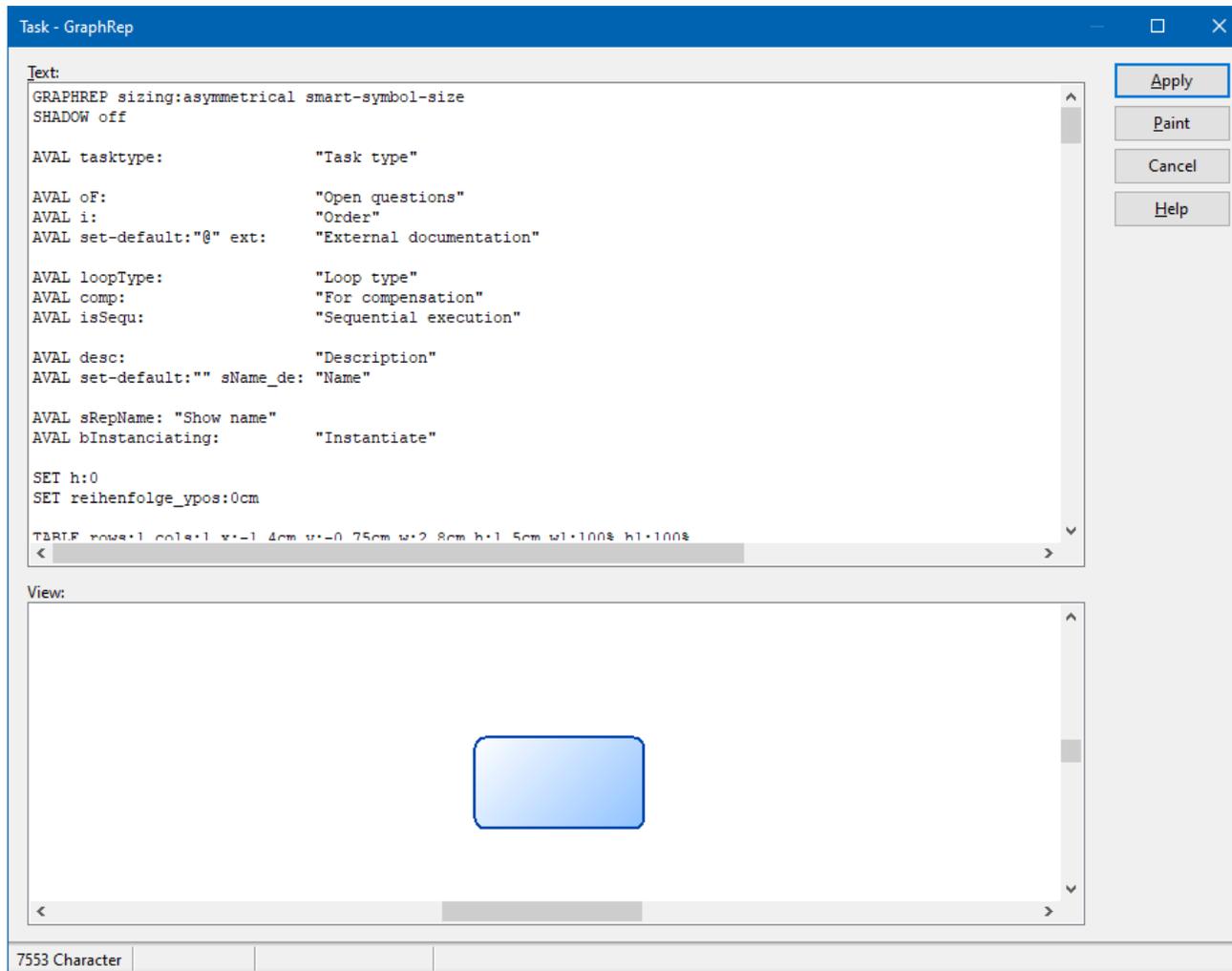
Task	
↳ _Conversion_	LONGSTRING (Long string)
↳ Aggregated costs	DOUBLE (Floating-point number)
↳ Aggregated execution time	TIME (Time)
↳ Aggregated personnel costs	DOUBLE (Floating-point number)
↳ Aggregated resting time	TIME (Time)
↳ Aggregated transport time	TIME (Time)
↳ Aggregated waiting time	TIME (Time)
↳ AnimRep (Metamodel)	STRING (Short string)
↳ Assignments (Metamodel)	RECORD (Record table)
↳ AttrRep (Metamodel)	LONGSTRING (Long string)
↳ Auditing	ENUMERATION (Enumeration)
↳ Average number of participants (Metamodel)	INTEGER (Integer)
↳ Beschreibung	STRING (Short string)
↳ Bezeichnung	STRING (Short string)
↳ Call activity	INTERREF (Inter-model reference)
↳ Cardinality	STRING (Short string)
↳ Categories (Metamodel)	STRING (Short string)
↳ Class cardinality (Metamodel)	STRING (Short string)
↳ ClassAbstract	INTEGER (Integer)
↳ Classification	ENUMERATIONLIST (Enumeration list)
↳ ClassName	STRING (Short string)
↳ ClassVisible	INTEGER (Integer)
↳ Collection	ENUMERATION (Enumeration)
↳ Comment	STRING (Short string)
↳ Completion condition	STRING (Short string)
↳ Continuous execution (Metamodel)	ENUMERATION (Enumeration)
↳ Cooperation mode (Metamodel)	ENUMERATION (Enumeration)
↳ Cooperative (Metamodel)	ENUMERATION (Enumeration)
↳ Costs	DOUBLE (Floating-point number)
↳ Description	STRING (Short string)
↳ Display responsible role	ENUMERATION (Enumeration)
↳ Documentation (Metamodel)	STRING (Short string)
↳ Doku	STRING (Short string)
↳ DokuSim	STRING (Short string)
↳ Done by (Metamodel)	STRING (Short string)
↳ EDP batch costs	DOUBLE (Floating-point number)
↳ EDP transaction costs	DOUBLE (Floating-point number)
↳ Execution interruptable (Metamodel)	ENUMERATION (Enumeration)
↳ Execution time (Metamodel)	TIME (Time)
↳ External documentation	PROGRAMCALL (Program call)
↳ External tool coupling (Metamodel)	STRING (Short string)
↳ fontcolor (Metamodel)	EXPRESSION (Expression)
↳ For compensation	ENUMERATION (Enumeration)
↳ Global task	ENUMERATION (Enumeration)
↳ GraphRep (Metamodel)	LONGSTRING (Long string)
↳ HlpTxt (Metamodel)	STRING (Short string)
↳ Id	EXPRESSION (Expression)
↳ Info on results	STRING (Short string)

New ▾
 Edit...
 Copy...
 Delete
 View ▾
 Close
 Help



Notation

GraphRep: A script language for the graphical representation



Appearance of Classes in the Modelling Toolkit

The screenshot displays the Modelling Toolkit interface. On the left, the 'Explorer - Model groups' pane shows a 'Business process diagram (BPMN 2.0) - new' model. The central workspace shows a BPMN diagram with a yellow circle connected to a red rectangle labeled 'Send invoice'. A vertical toolbar on the left contains various symbols, with a red oval highlighting a group of symbols labeled 'Classes' and another red oval highlighting a group of symbols labeled 'Relations classes'. On the right, the 'Send invoice (Task)' configuration panel is open, with a red oval highlighting its content labeled 'Attributes'. The configuration panel includes fields for Name, Show name, Task type, Order, Description, Comment, Open questions, and Id.

Classes

Relations classes

Attributes

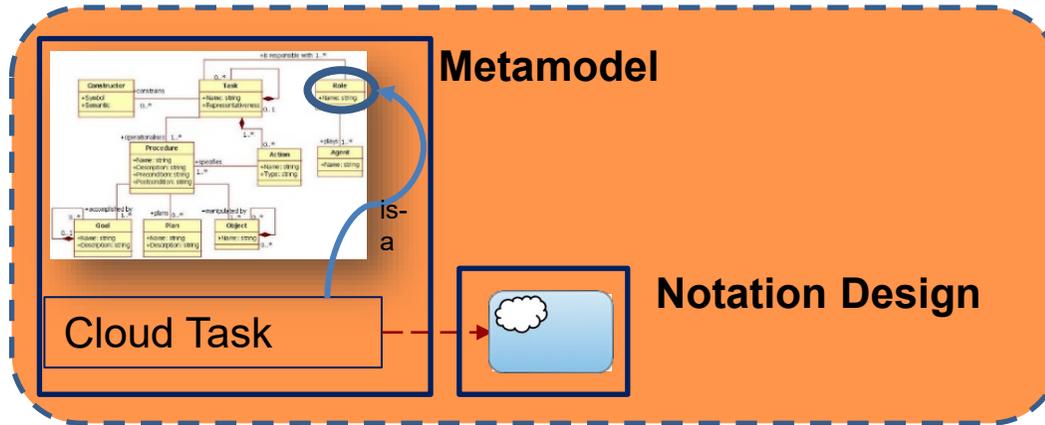


Change of Metamodel

- Example: new task type Cloud Task



Metamodel Engineer



Meta-modeling

Feedback
Amendments
Improvements

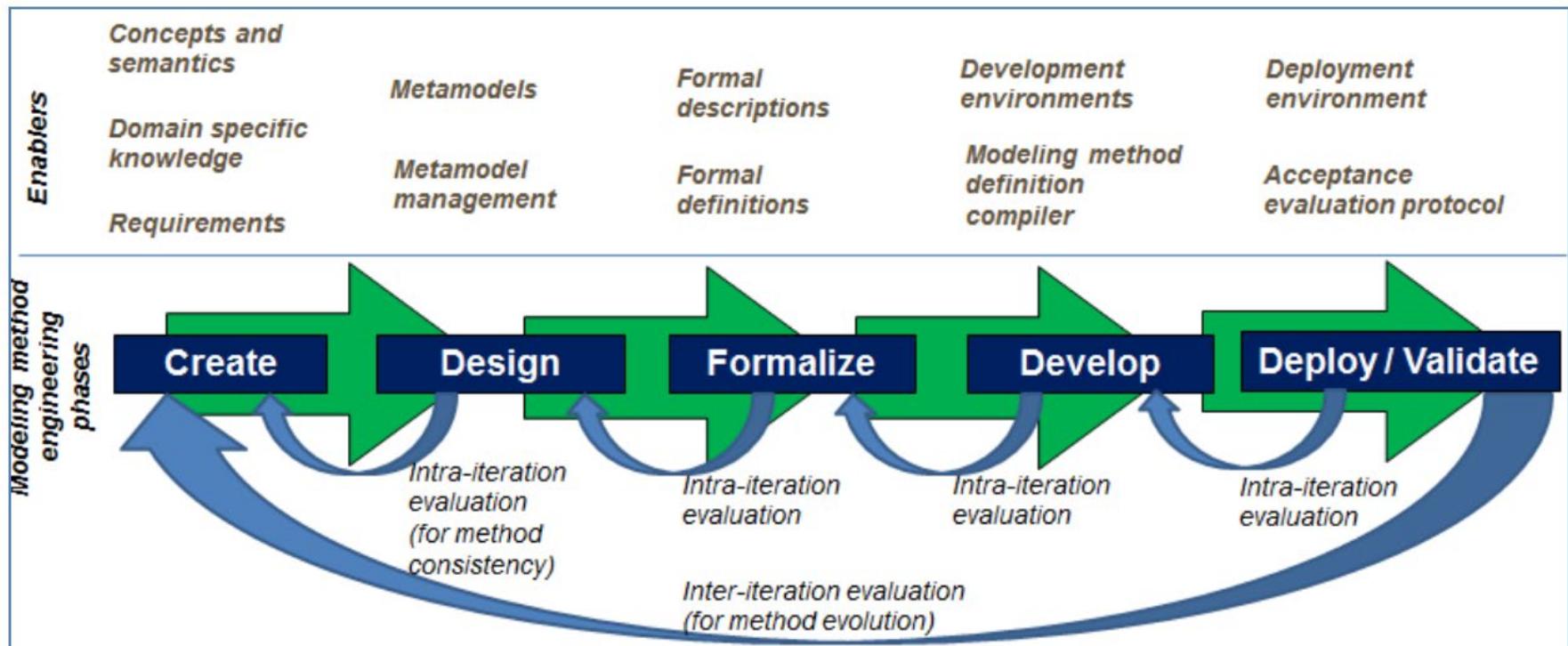
Modeler



Modeling

The AMME LifeCycle

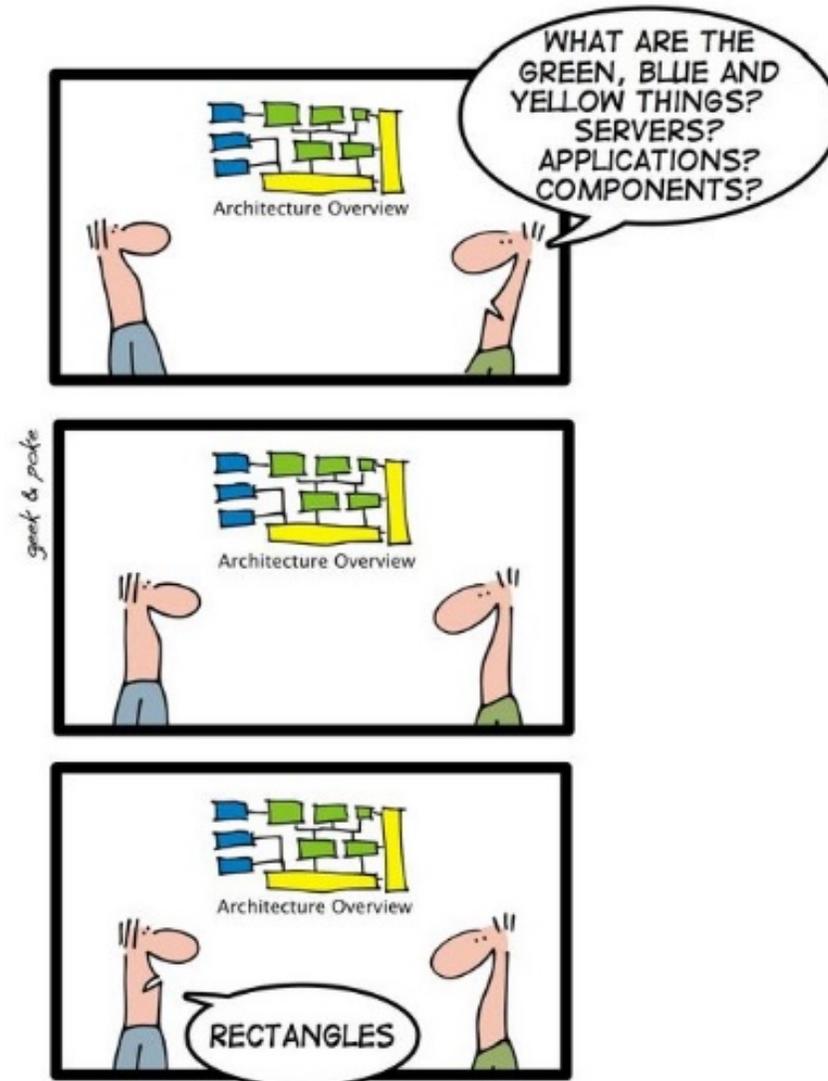
Agile Modeling Method Engineering



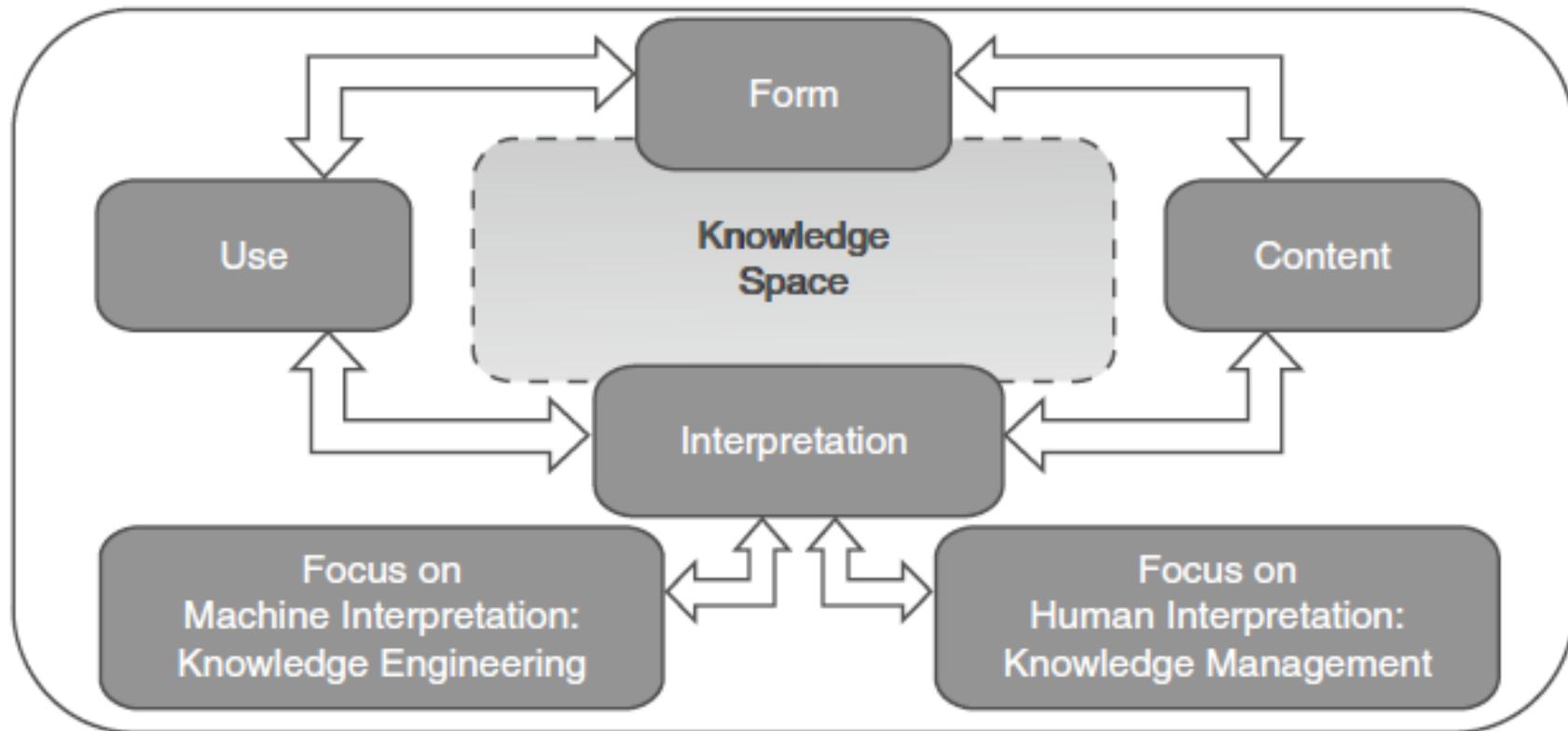
(Karagiannis 2015)

Knowledge in Models

Interpretation of Models

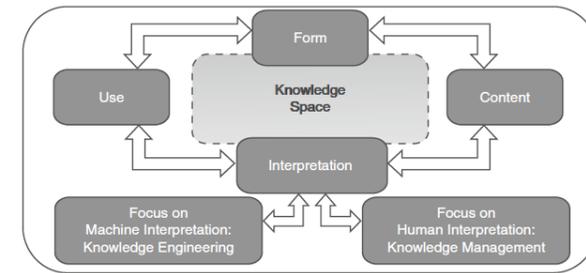


Dimensions of a Knowledge Space



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

- **Use:** Stakeholders and their concerns determine the relevant subset of the knowledge

Form: modeling language



- **Form:** Syntax and semantic of **modeling language elements.**

Content: Instantiation of the model elements



- **Content:** **Domain** in which knowledge engineering is applied, is represented in the labels

- **Interpretation:** Giving meaning to a model:

- ◆ Graphical models are cognitively adequate for human
- ◆ Machines need more formal representation

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 Component
- ◆ Domain: «ERP System» is business software

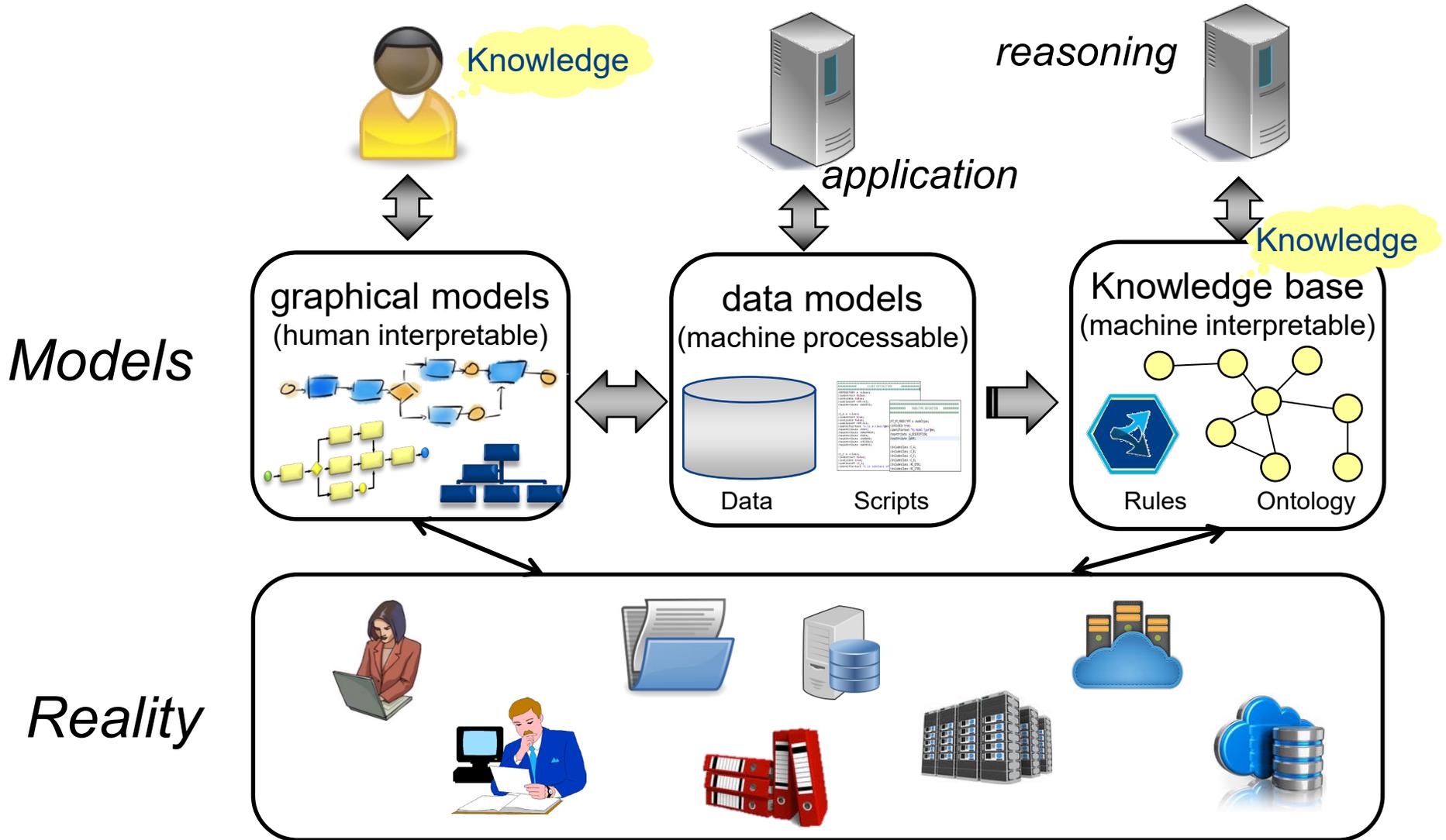


- ◆ Model element: Task
- ◆ Domain: «Cook pasta» is about preparing food

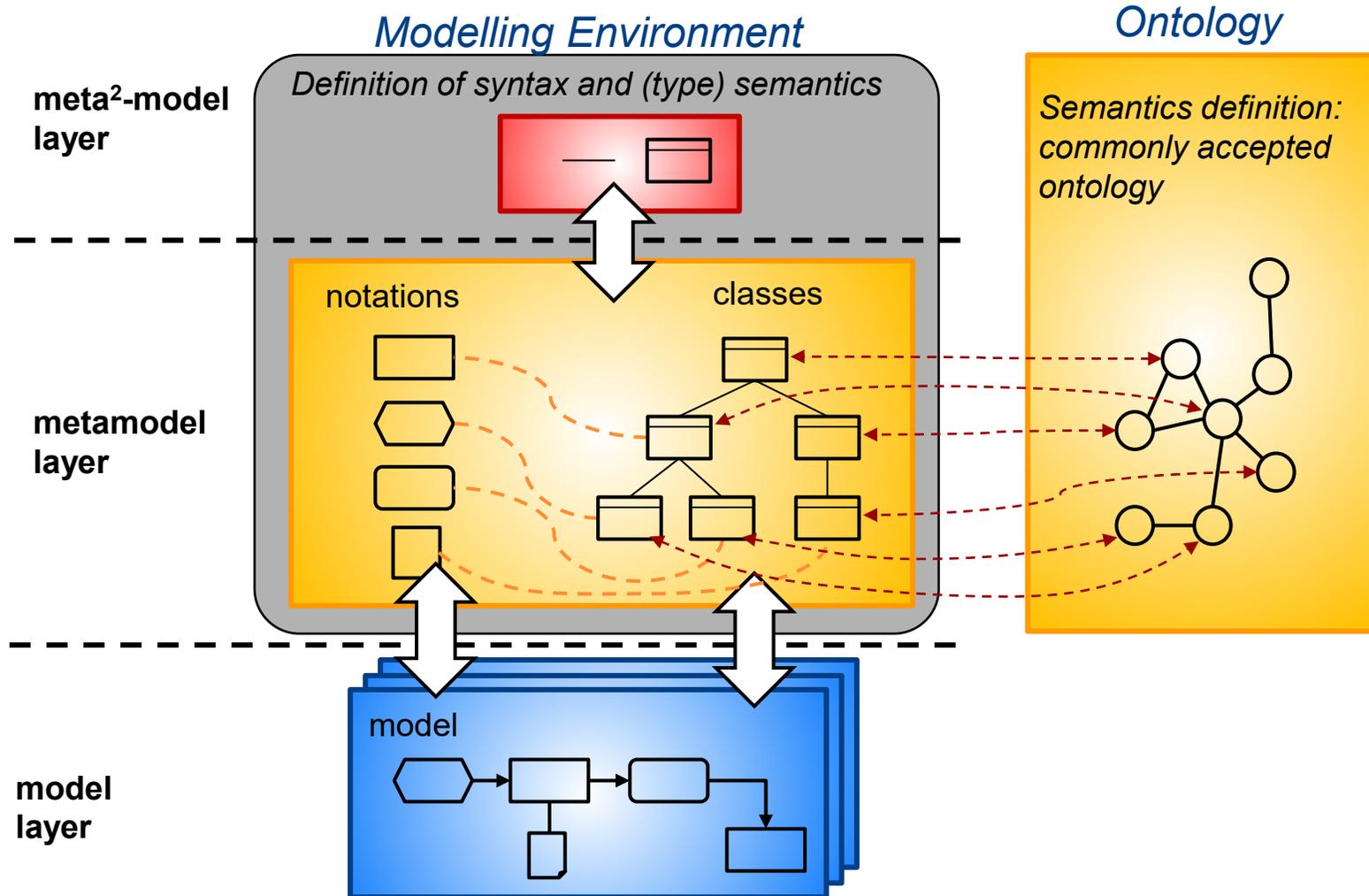
- The objective is to represent the knowledge so that it can be interpreted by a system for decision making and problem solving

Semantic Lifting

Semantic Lifting: Map Models into an Ontology



Semantic Lifting: Map Models into an Ontology



←- - - -> **ontological metamodelling (lifting): explication of type semantics**

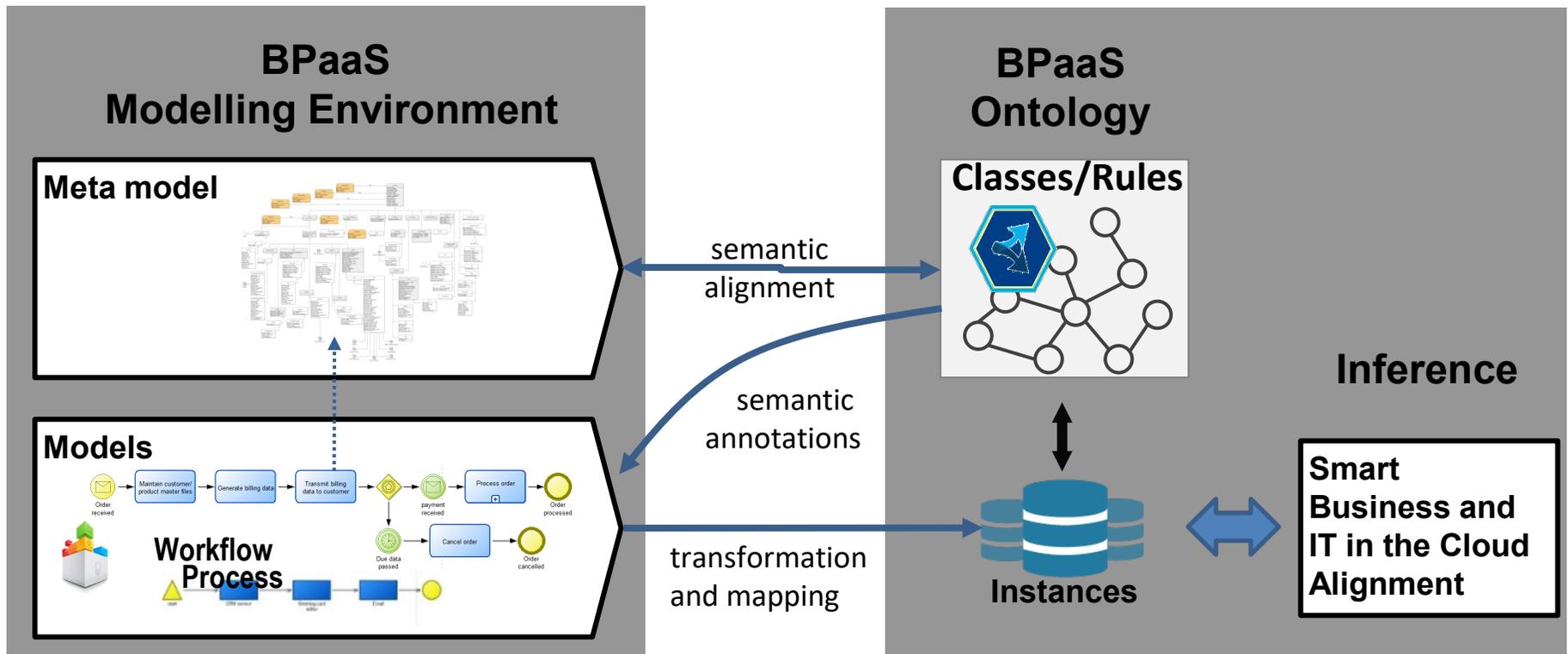
Semantic Lifting

- Map models into an ontology
 - ◆ Semantics: Classes of the metamodel are aligned with classes in the ontology
 - ◆ Interpretation: For each element in a model an instance of the ontology is created
 - ◆ Content: Model elements are annotated with domain knowledge from ontology
 - ◆ Inference of the ontology can be applied to the knowledge base

Example: Business Process as a Service

human interpretation
informal and semi-formal

machine interpretation
formal



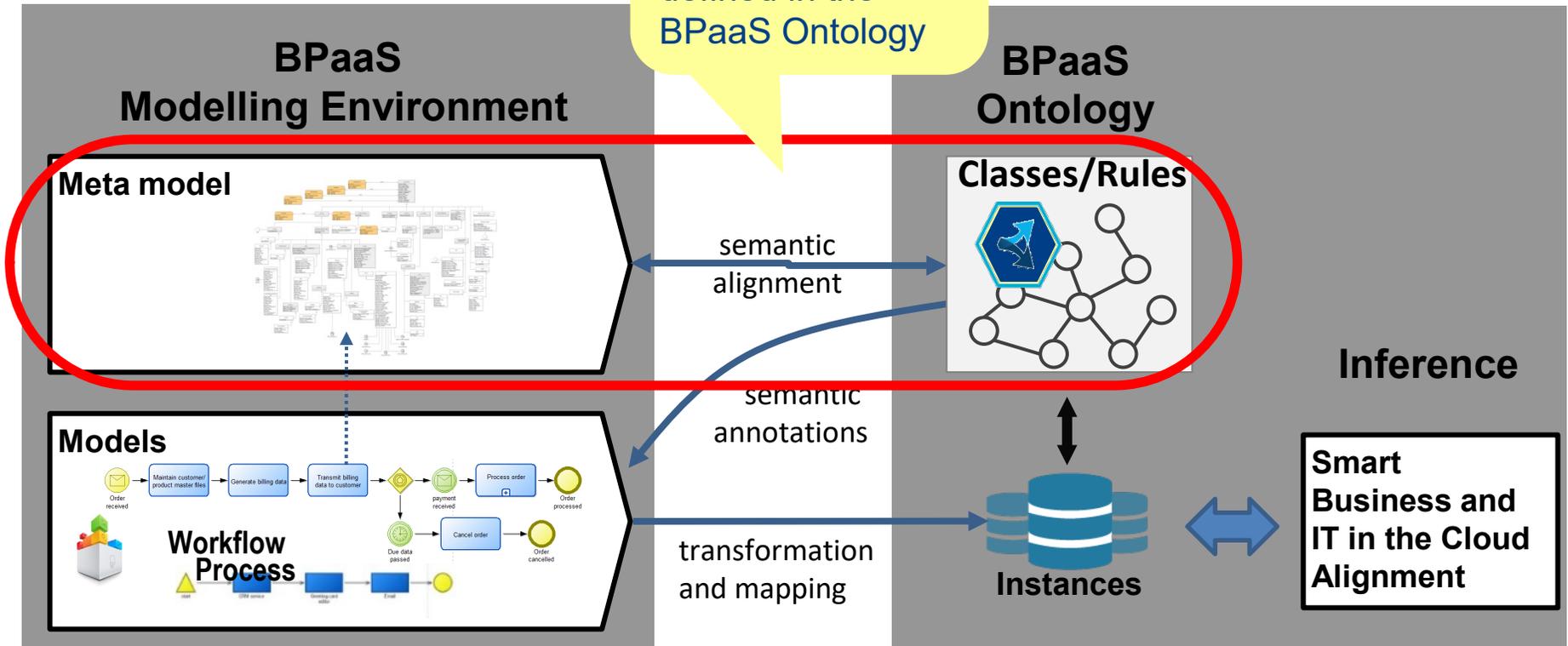
From: CoudSocket Project

Example: Business Process as a Service

human interpretation
informal and semi-formal

The semantics of the meta-model elements is defined in the BPaaS Ontology

machine interpretation
formal



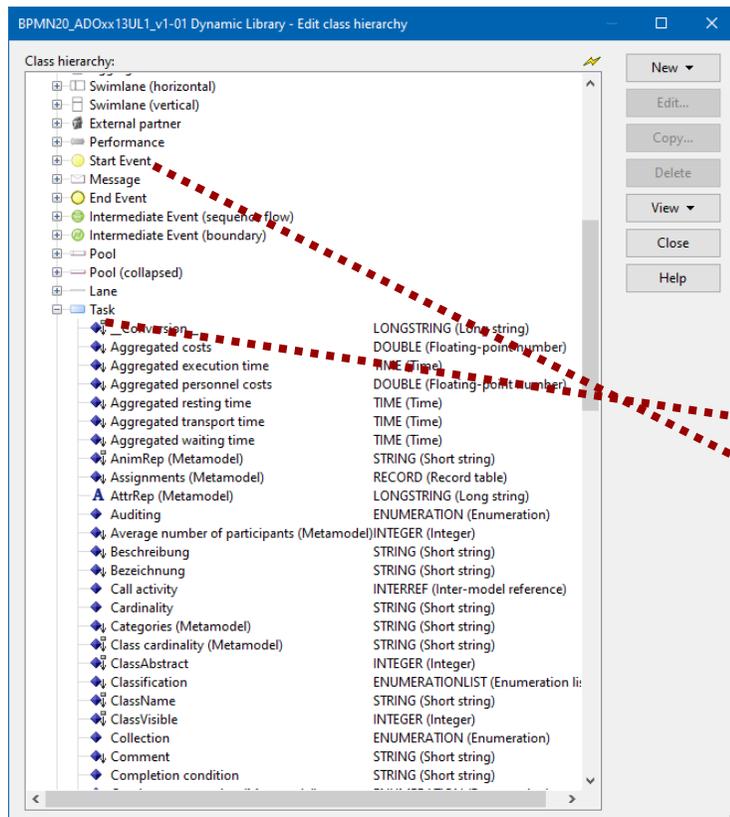
From: CoudSocket Project



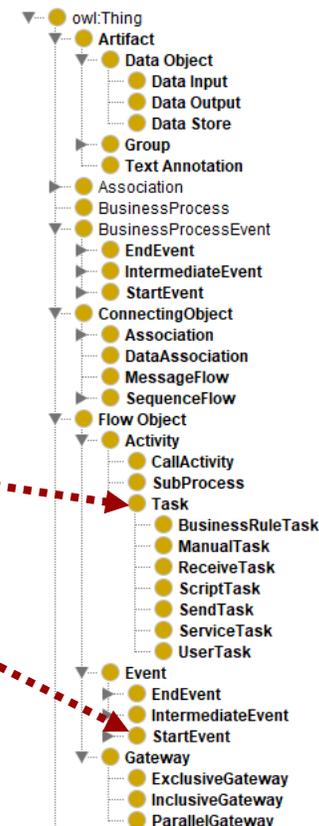
Semantic Alignment

The ontology contains classes for all modelling elements

BPMN Modelling Language in ADOxx

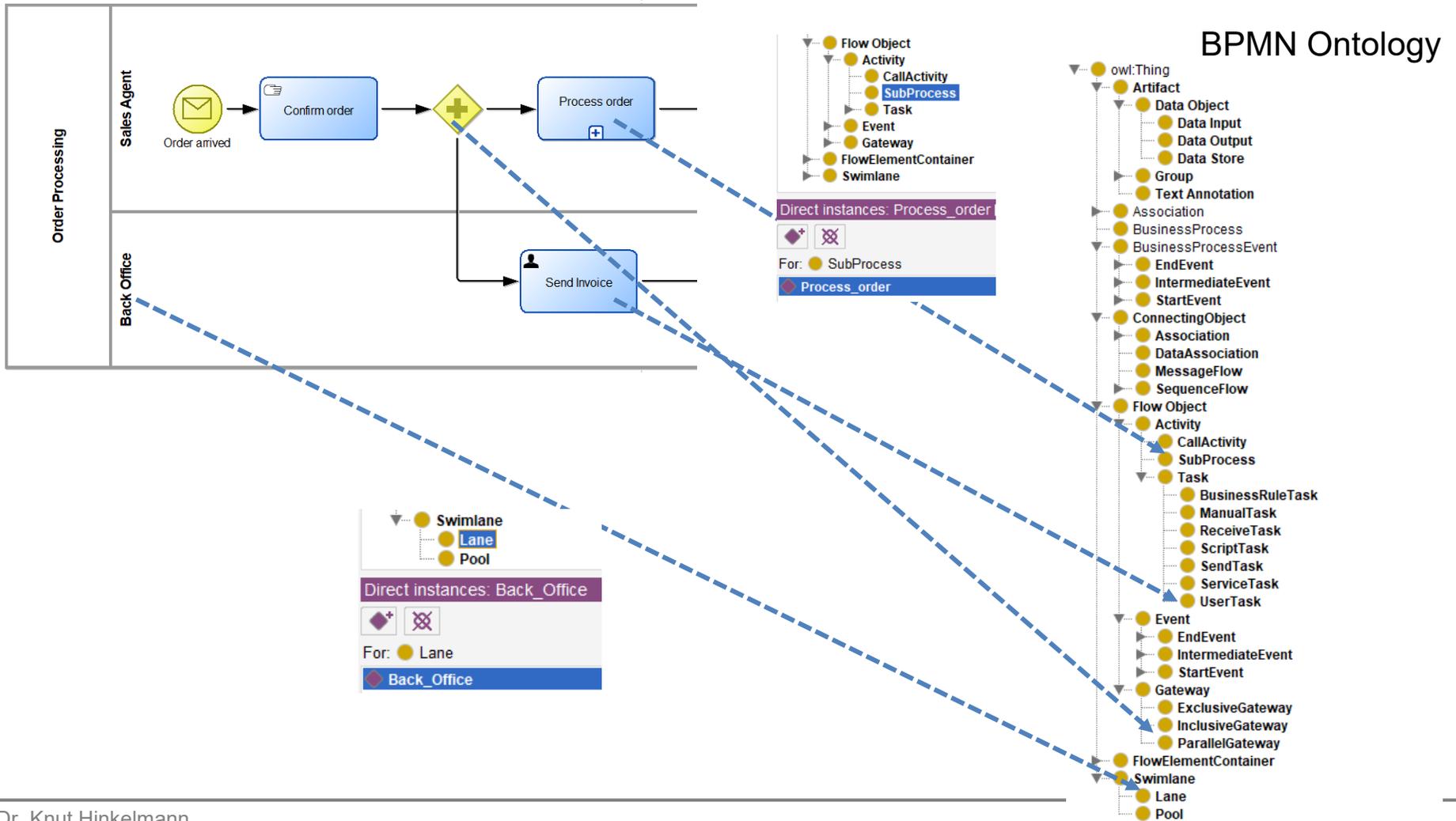


BPMN Ontology



Transformation and Mapping

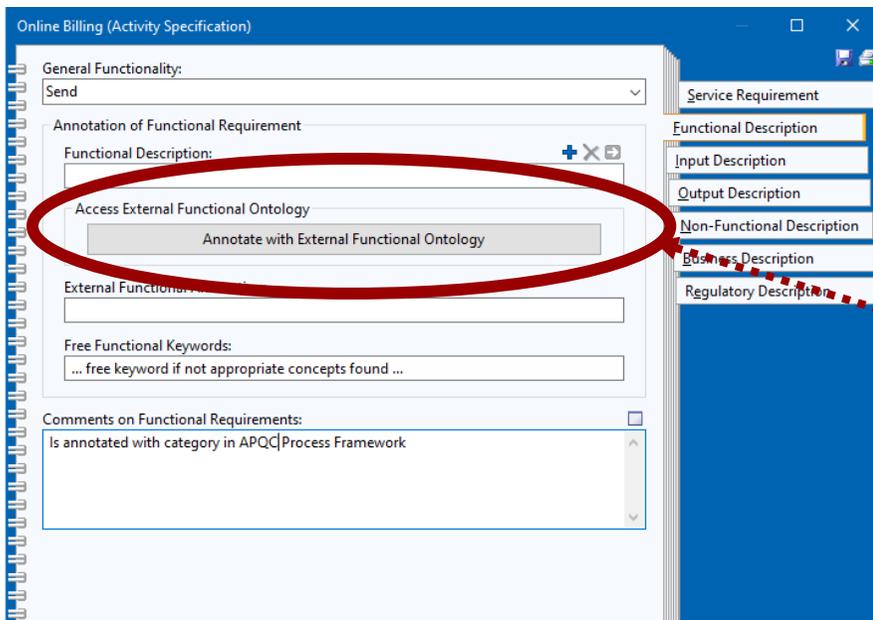
The model elements are exported as instances ontology classes



Semantic Annotations

Annotate modeling elements with classes from the domain ontology

Example: Functionality of a Service



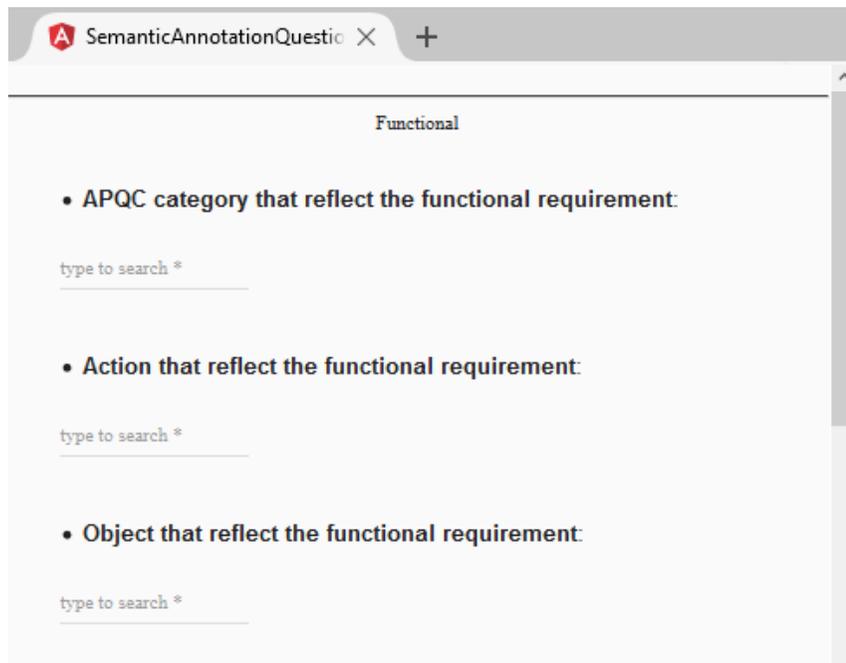
Domain Ontology:
APQC Process Classification Framework



Application Example for Semantic Lifting

Cloud Service Selection

Functionality

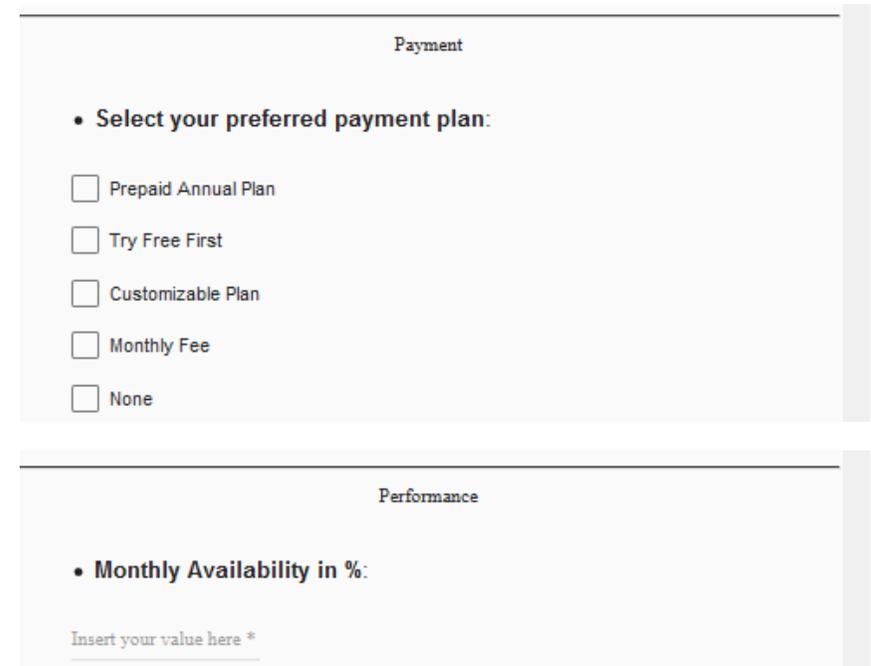


SemanticAnnotationQuestio X +

Functional

- APQC category that reflect the functional requirement:
type to search *
- Action that reflect the functional requirement:
type to search *
- Object that reflect the functional requirement:
type to search *

Non-functional requirements



Payment

- Select your preferred payment plan:
 - Prepaid Annual Plan
 - Try Free First
 - Customizable Plan
 - Monthly Fee
 - None

Performance

- Monthly Availability in %:
Insert your value here *

Drawbacks of Semantic Lifting

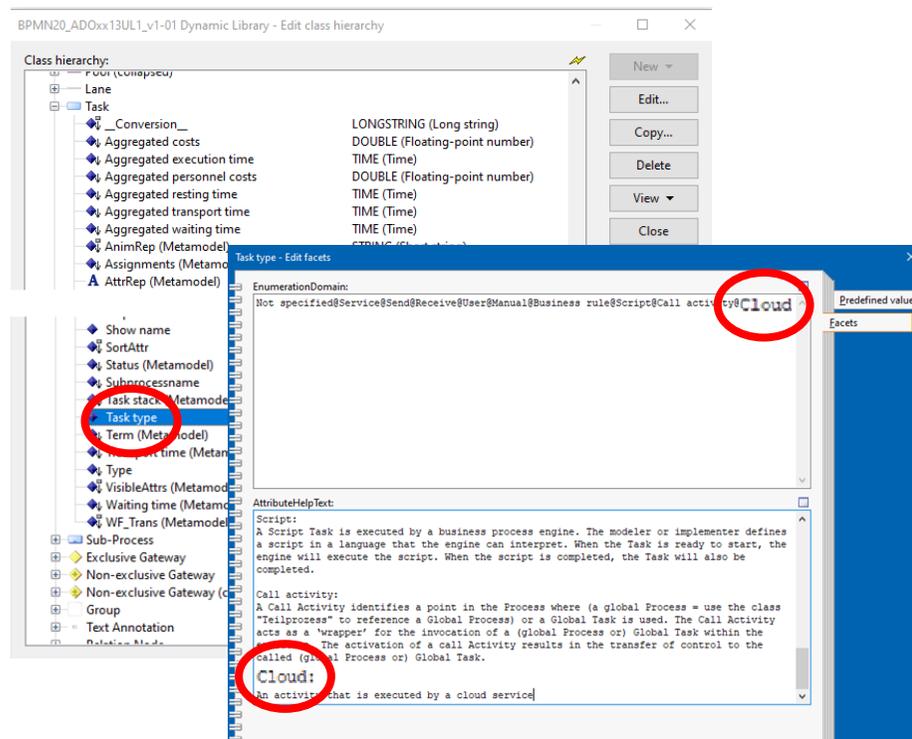
- Separate Environments for
 - ◆ Modelling
 - ◆ Knowledge Base (Inferencing)
- Inconsistency: Both metamodel and ontology must be aligned but are maintained independently:
 - ◆ Metamodel and ontology must represent the same semantics
 - ◆ Each change in metamodel must be reproduced in the ontology and vice versa
- Effort: After each change the models must be translated again into the ontology instances

Example: New Model Element

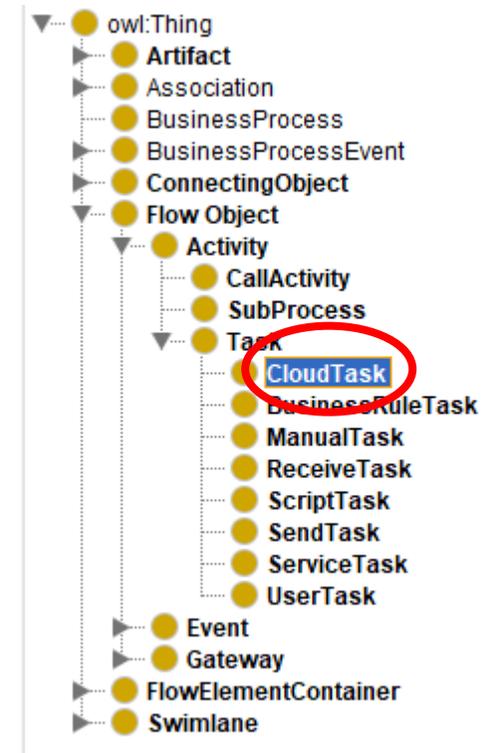
■ New task type: Cloud Task



Change in the meta model:

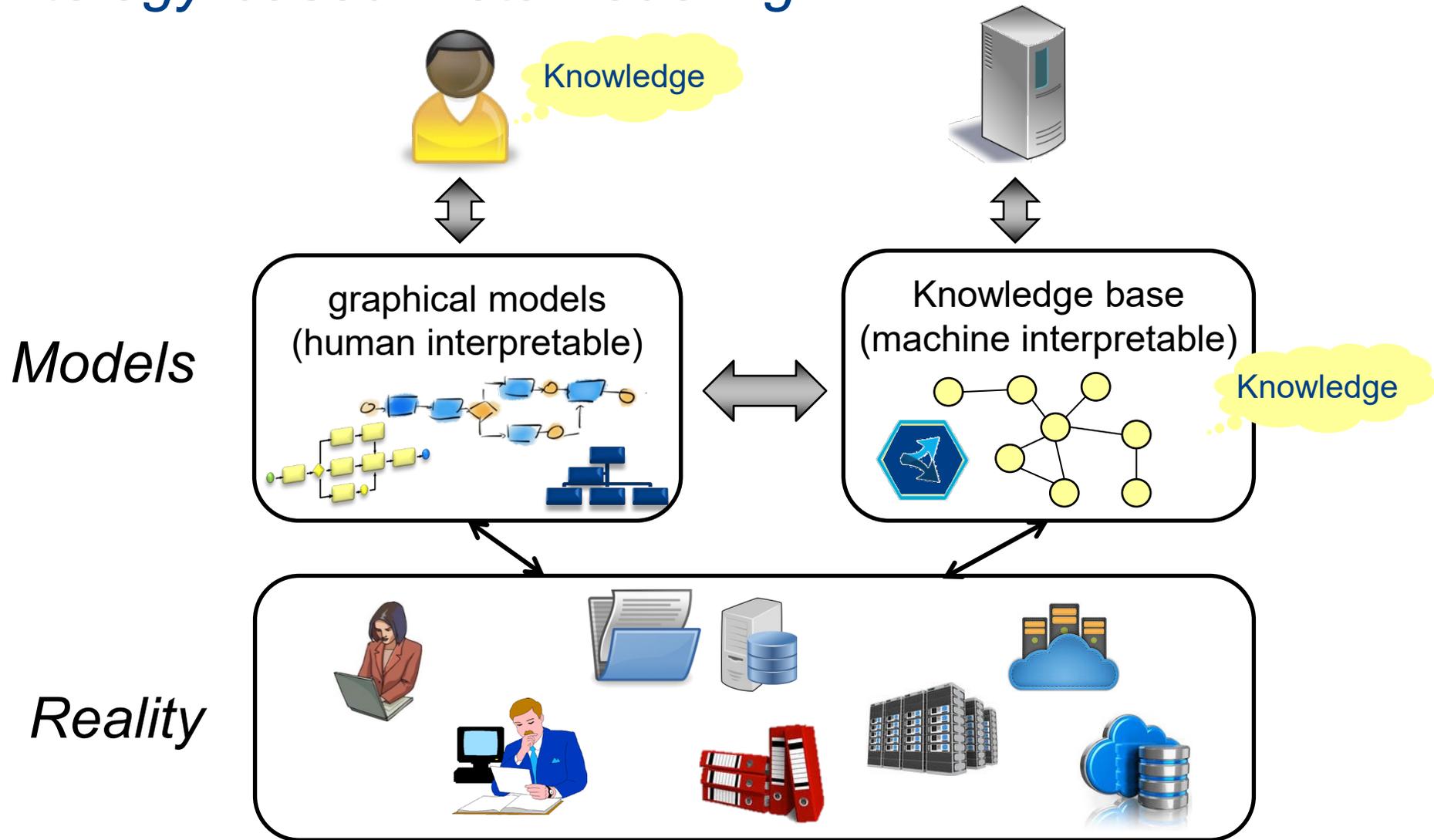


Change in the ontology:

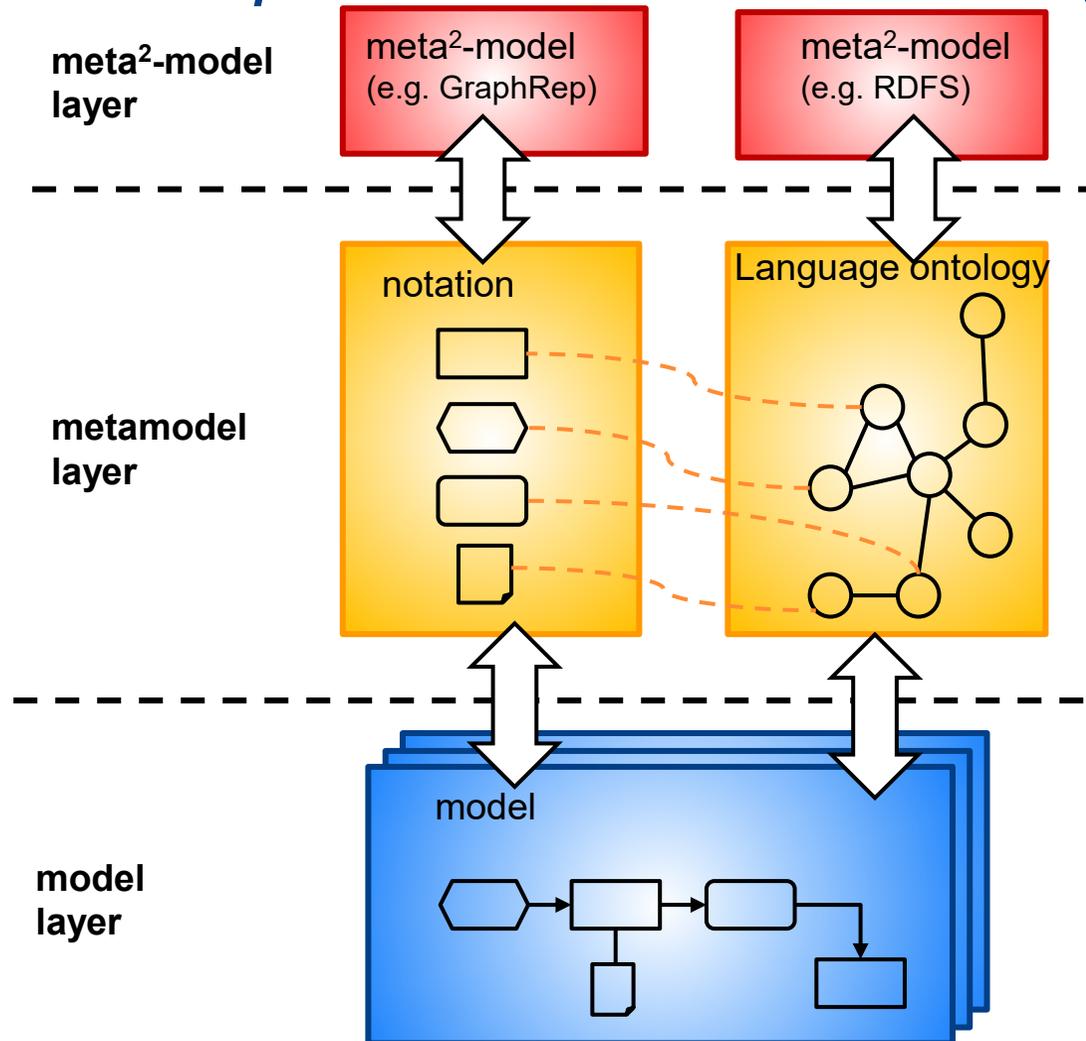


Ontology-based Metamodelling

Ontology-based Metamodeling

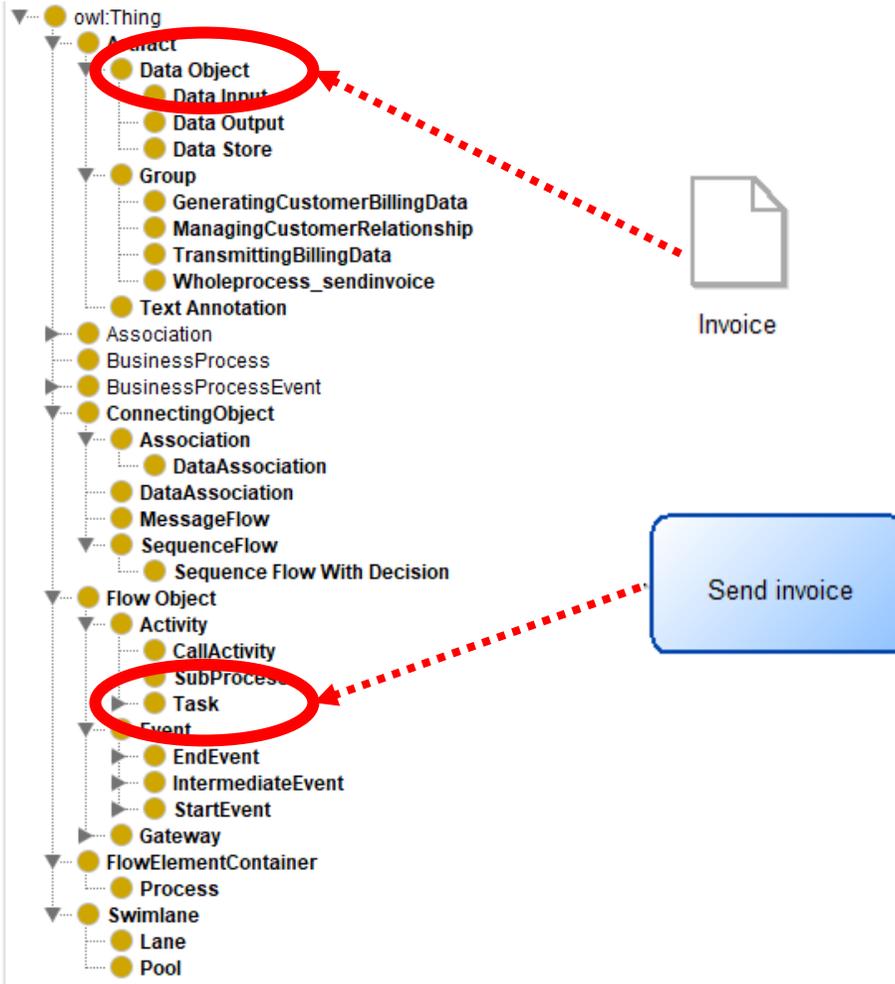


Ontology-based Metamodeling (1): Metamodel is represented as an Ontology

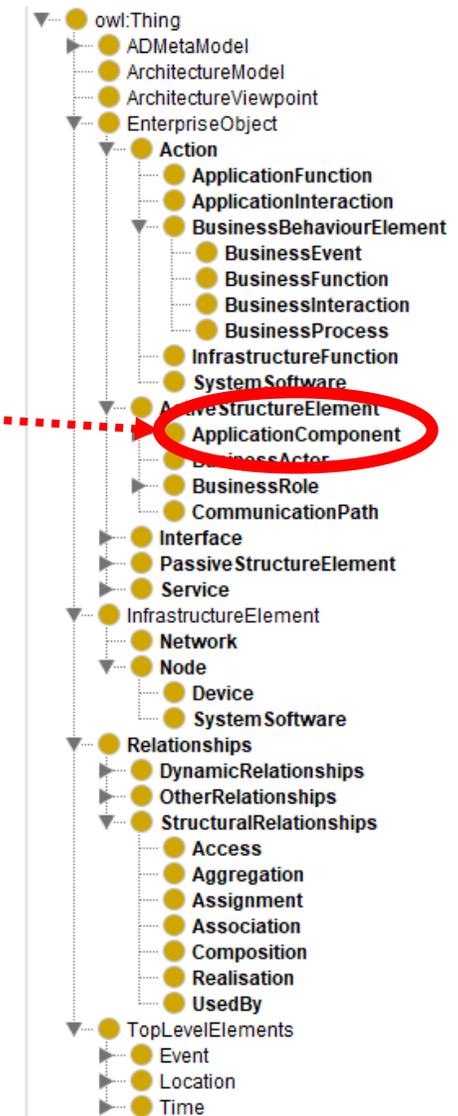


Modelling Language Ontologies

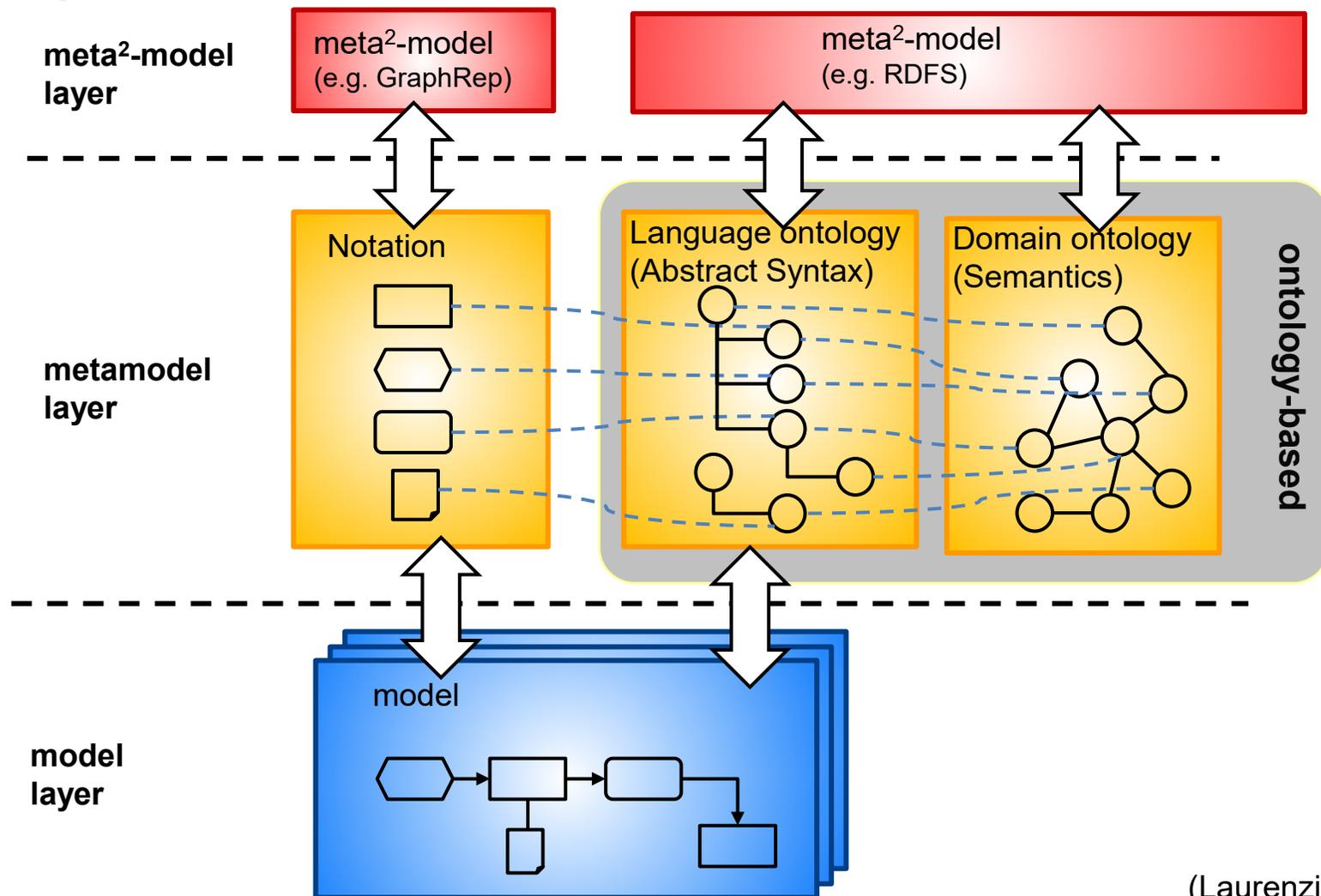
BPMN



Archimate



Ontology-based Metamodeling (2): Ontologies for Metamodel and Content

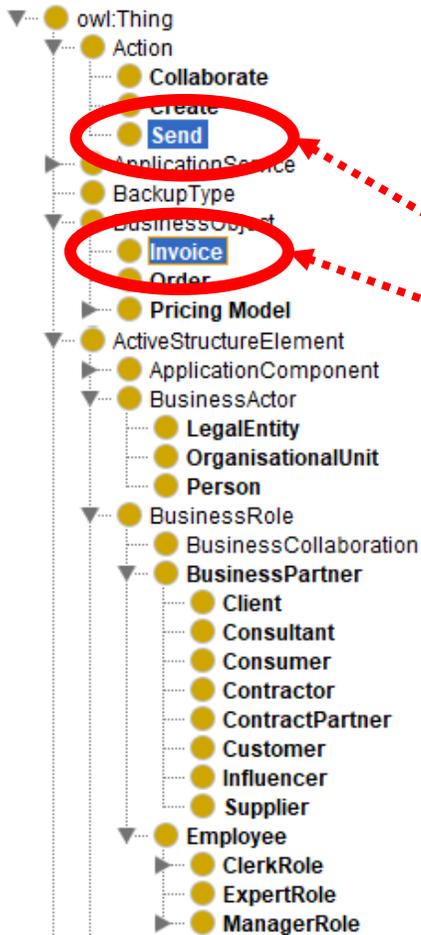


(Laurenzi et al. 2018)



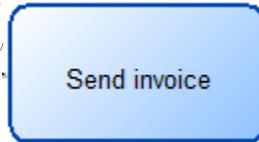
Domain Ontologies

Enterprise Ontology (excerpt)



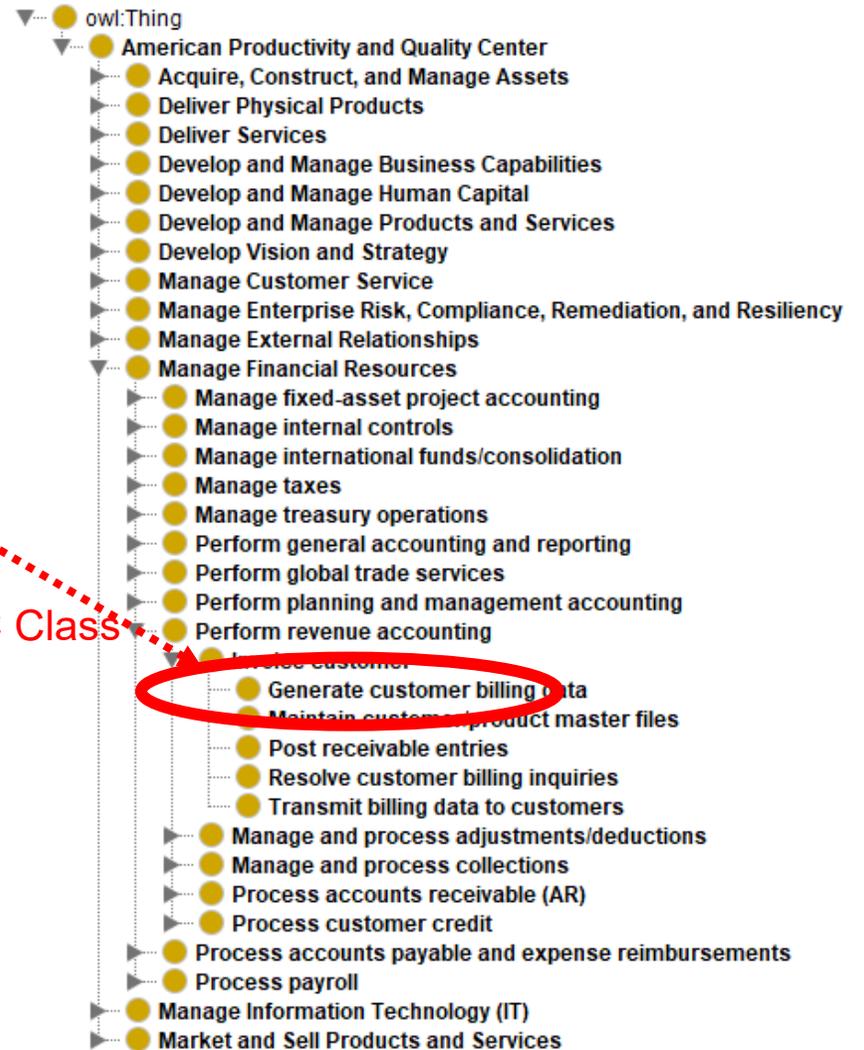
Action type

Object



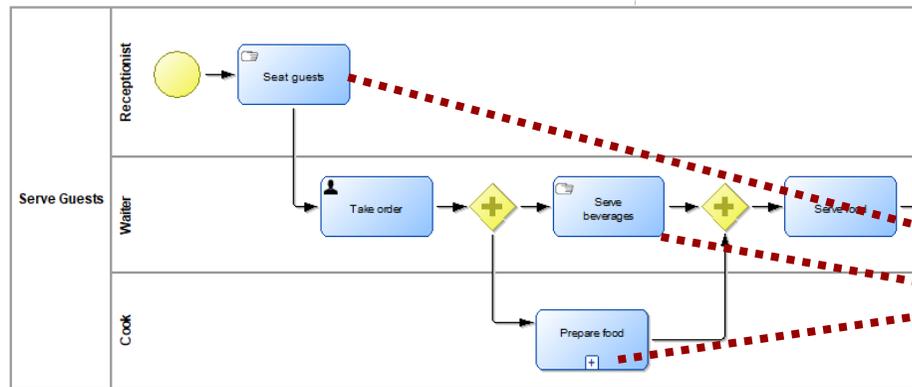
APQC Class

Domain Ontology: APQC Process Classification Framework



Ontology-Based Modeling

- Single environment for modelling and ontology
- Model elements are directly created as instances in the ontology



Class hierarchy: ManualTask Asserted

- owl:Thing
 - Artifact
 - Association
 - BusinessProcess
 - BusinessProcessEvent
 - ConnectingObject
 - Flow Object
 - Activity
 - CallActivity
 - SubProcess
 - Task
 - BusinessRuleTask
 - ManualTask**
 - ReceiveTask
 - ScriptTask
 - SendTask
 - ServiceTask
 - UserTask
- Event
- Gateway
- FlowElementContainer
- Swimlane
 - Lane
 - Pool

Individuals

- ◆ Cook
- ◆ Prepare_Food
- ◆ Receptionist
- ◆ Seat_guests
- ◆ Serve_Beverages
- ◆ Serve_food
- ◆ Take_order
- ◆ Waiter

Agile Meta-Modeling

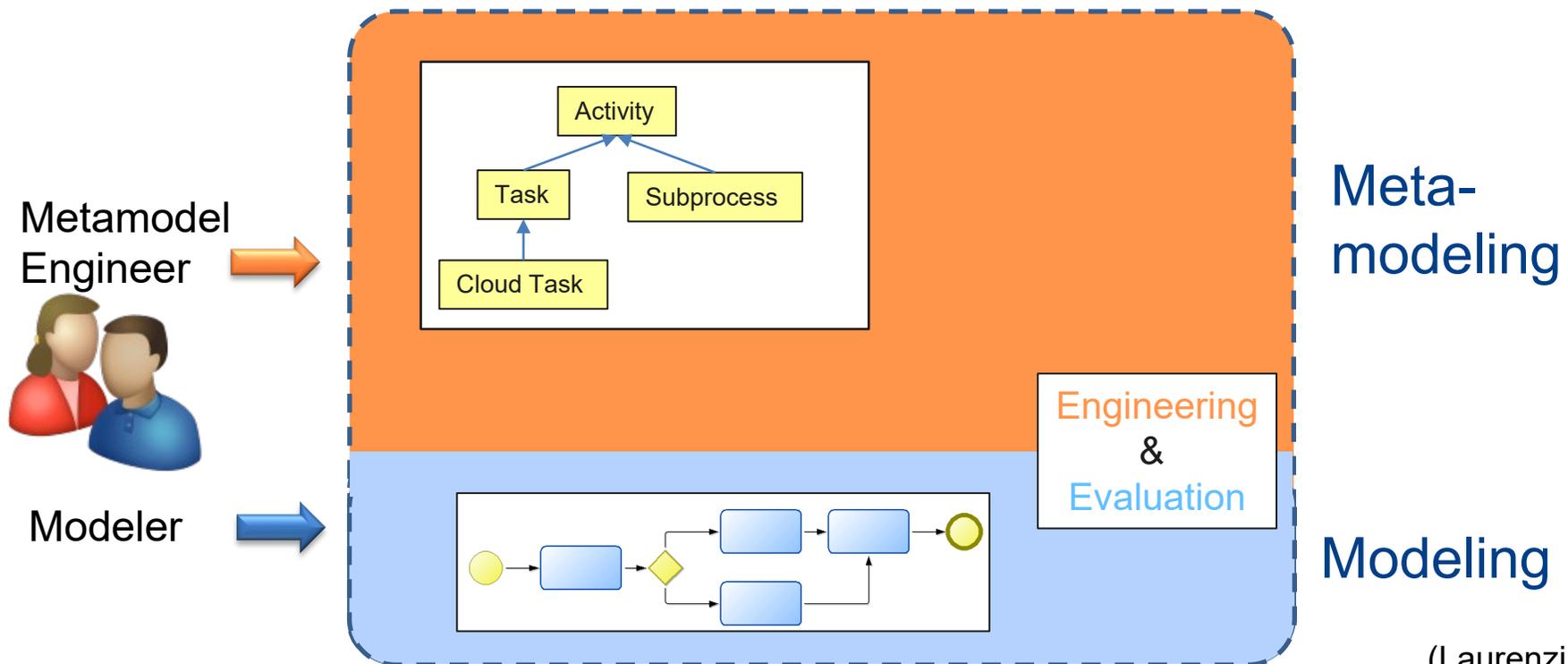


Objective

Adapt modeling languages and ensure a precise shared interpretation of new modeling constructs to both **humans and machines**

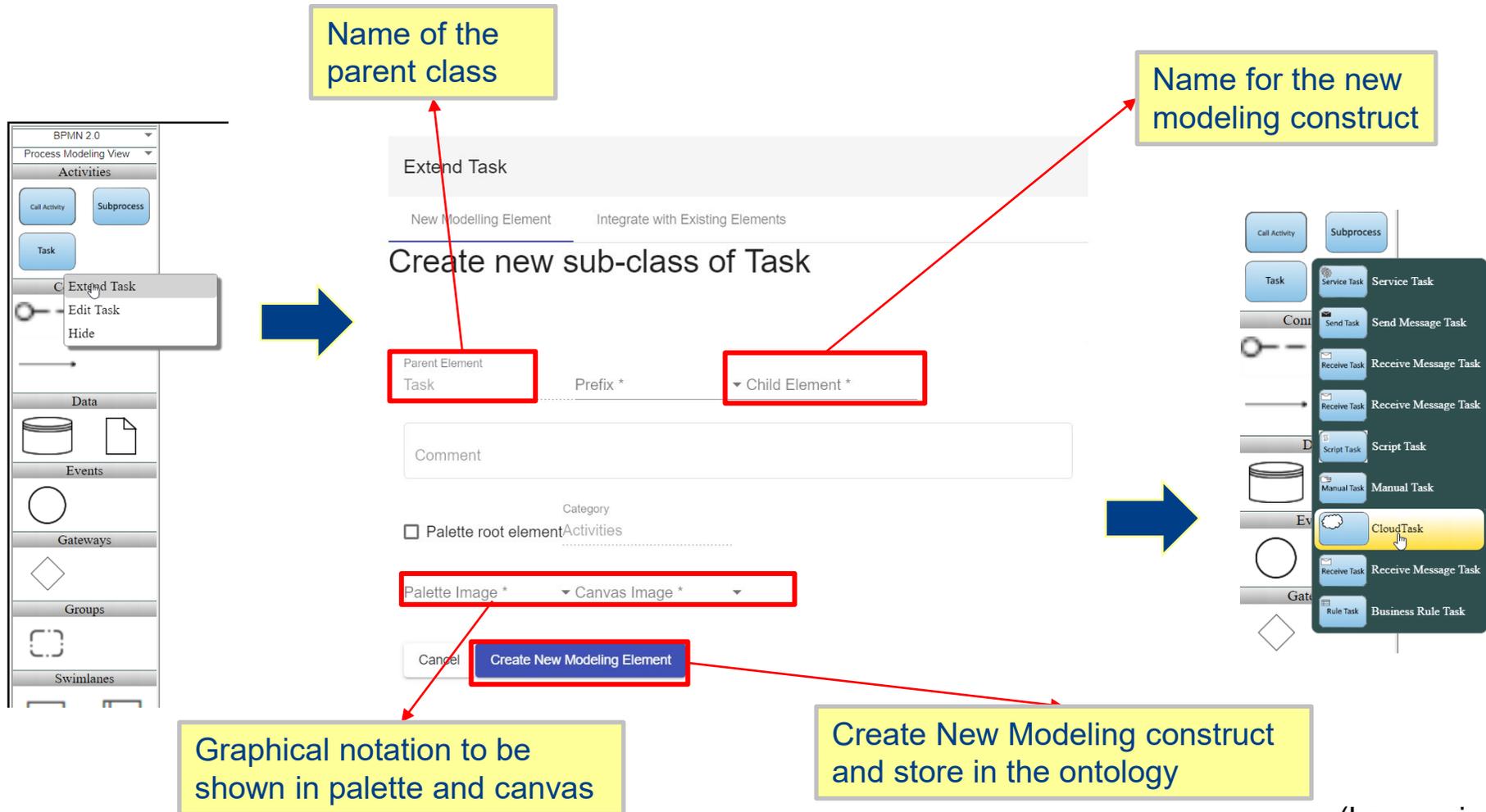
Integration Modeling and Metamodeling in a Single Environment

- Modeling and metamodeling in a single environment
- Tight collaboration between metamodel developer and modeler
- Modeler can also take the role of metamodel developer



(Laurenzi et al. 2018)

Integration of Meta-modeling and Modeling: On-the-fly Modeling Language Adaptation



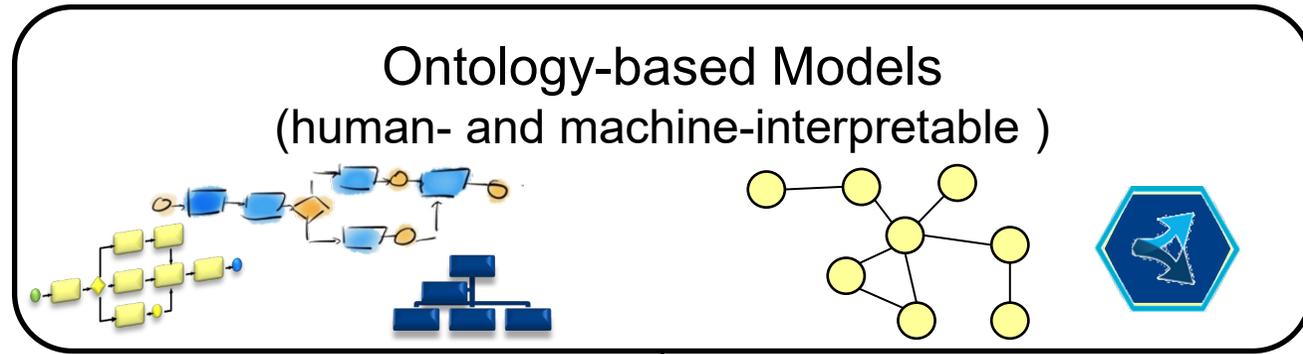
(Laurenzi et al. 2018)



Agile and Ontology-Aided Modeling Environment (AOAME)



*Models +
Knowledge*



Reality

