University of Applied Sciences and Arts Northwestern Switzerland
School of Business

Towards An Assistive and Pattern Learning-driven Process Modeling Approach

Emanuele Laurenzi, Knut Hinkelmann, Stephan Jüngling,
Devid Montecchiari, Charuta Pande and Andreas Martin

Business Reengineering

high-pace of change

high competition

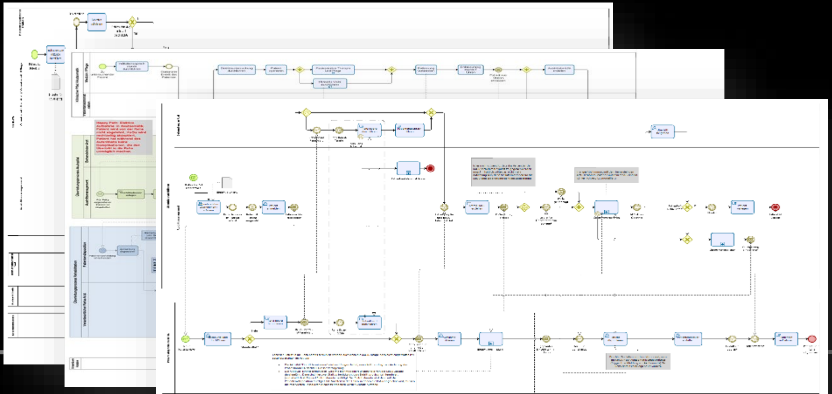
On demand customization

Fast delivery

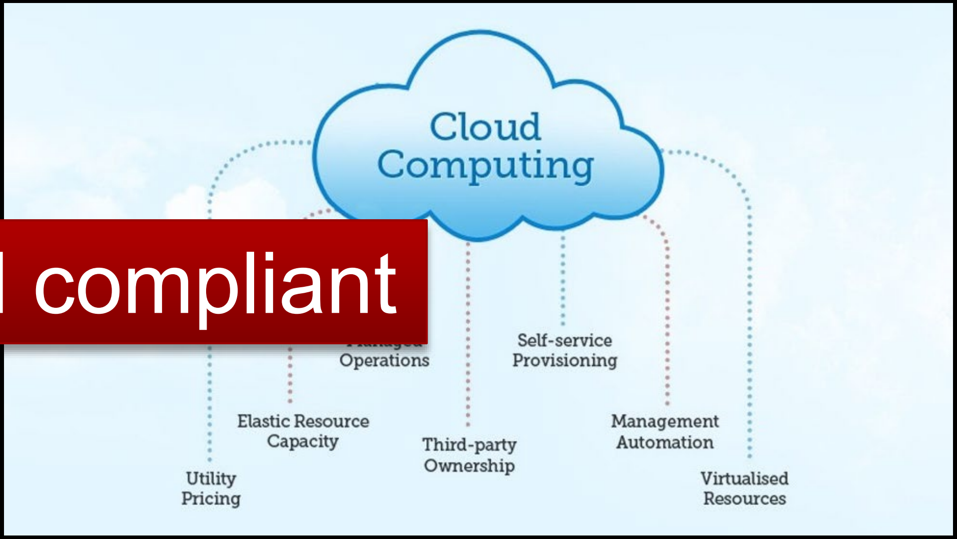
Continuous Acquisitions

Data-driven Economy

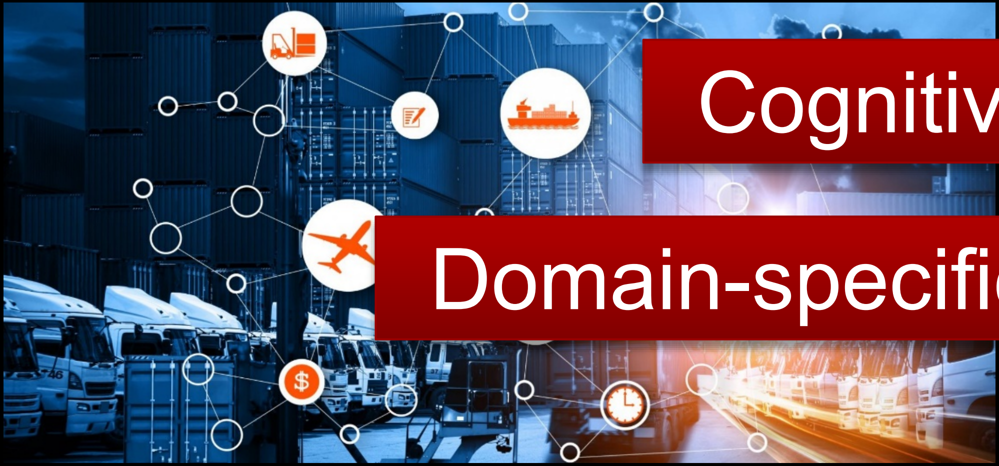
Change of policies and regulations



Challenge in Process Modeling



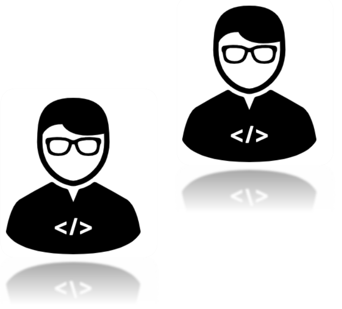
Standard compliant



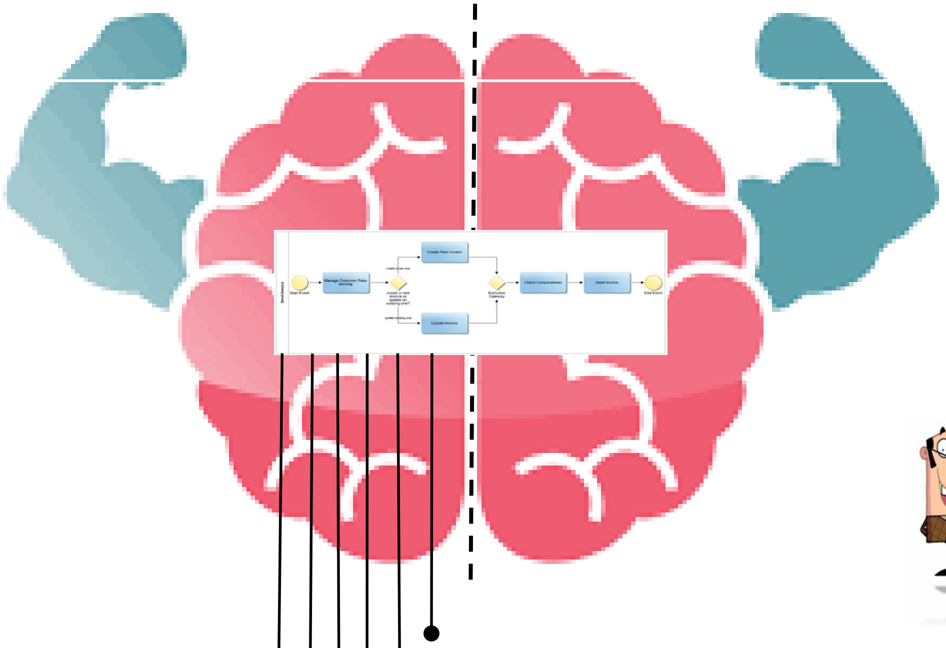
Cognitive adequate

Domain-specific inherent semantic

Design of a “Good” Process Models



Modeling Experts



Cognitive Adequate

Conform to modeling style and conventions

Appropriate terminology

Useful level of abstraction

Semantically correct

Syntactically correct

Domain Experts

Problems: (1) High Engineering Effort! (2) Process Models evolve!

Our Solution

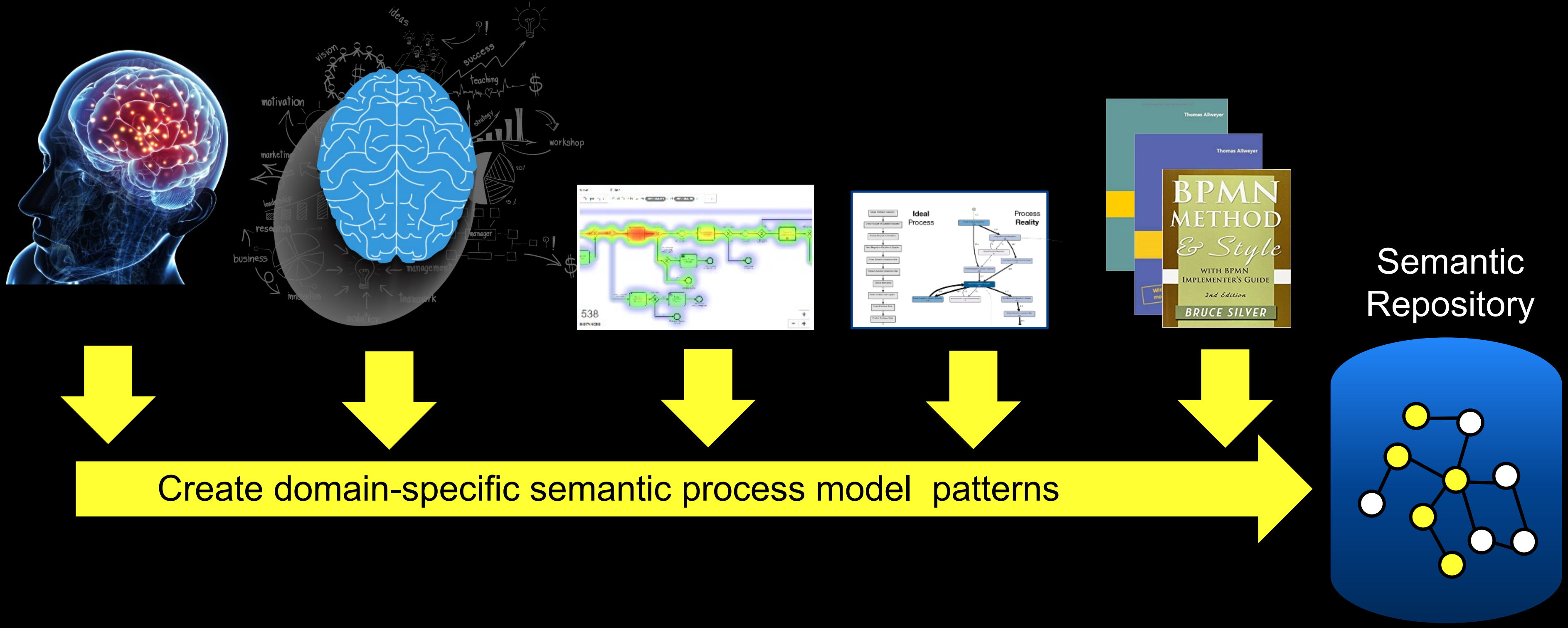
- Tackling the two major problems
  - Suggesting and Evolving

    Domain-Specific Semantic Process Model Patterns

Definition

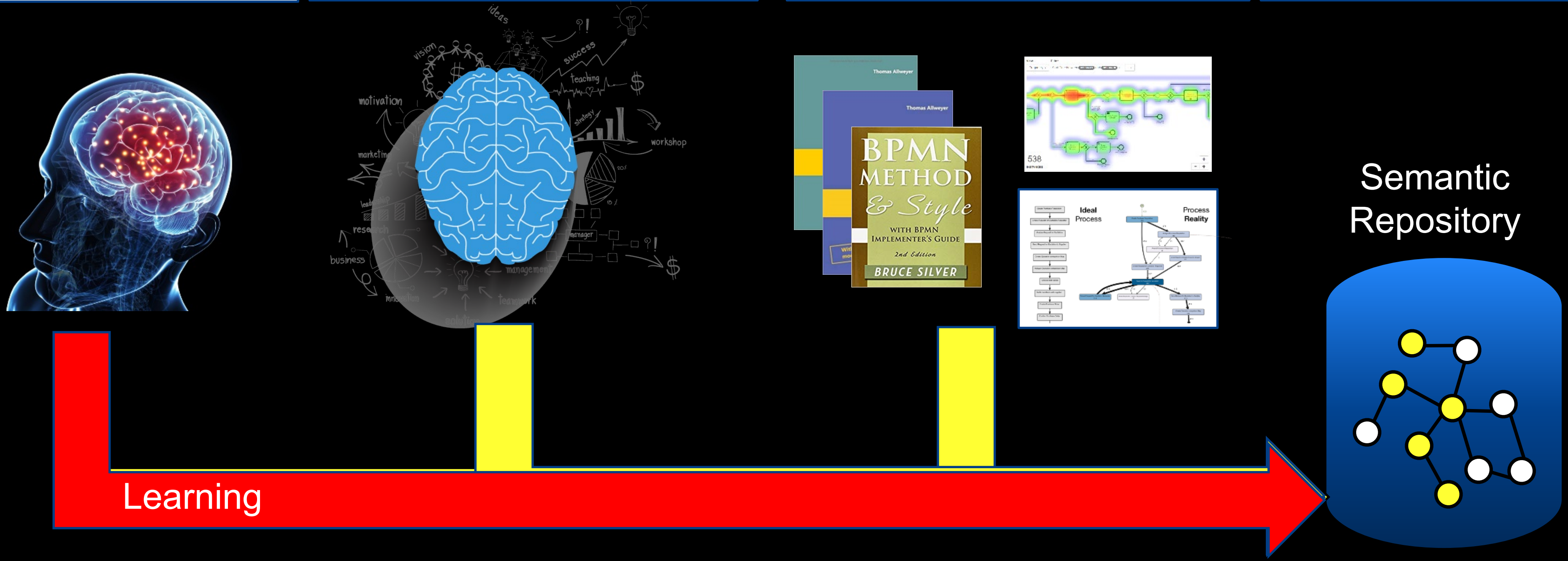
- A Domain-Specific Semantic Process Model Pattern is
  - A proven solution to a recurring problem that is related to the **creation or modification** of business process models in a specific context.
  - This solution consists of an **ontology-based representation** of either an end-to-end process model or a fragment of a process model.
  - The contains **relevant notions** for which a pattern can be labeled as “Good” or “Appropriate”.

Learning Sources



Mapping to the Knowledge Management

Internal Knowledge		External Knowledge	
Tacit Knowledge (in domain experts' heads)	Self-Aware Knowledge (in modeling experts' heads)	Documented Knowledge (in documents/databases)	Formal Knowledge (in knowledge bases)



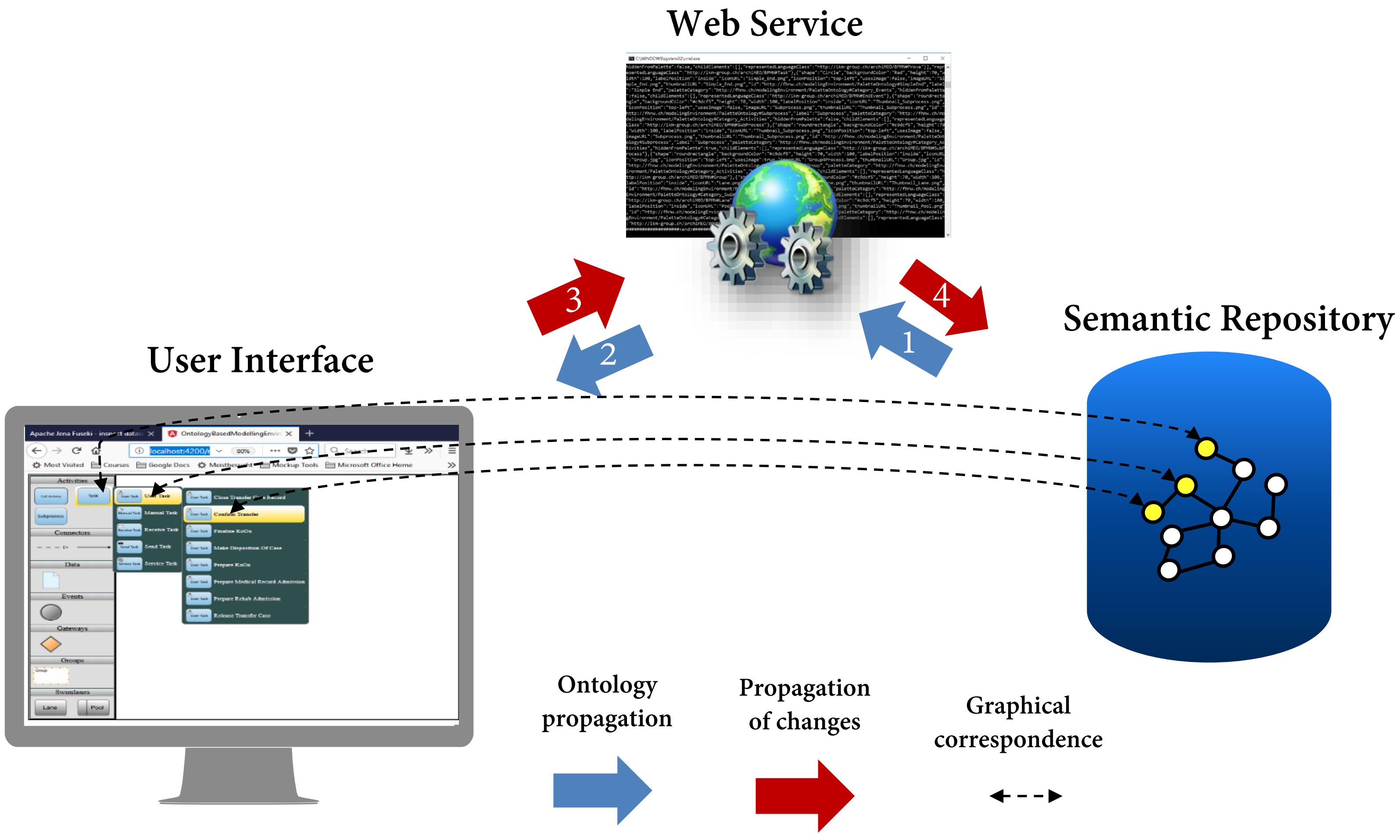
Research Direction
Learning and Engineering - A Combined Approach

■ 3-Phase Approach

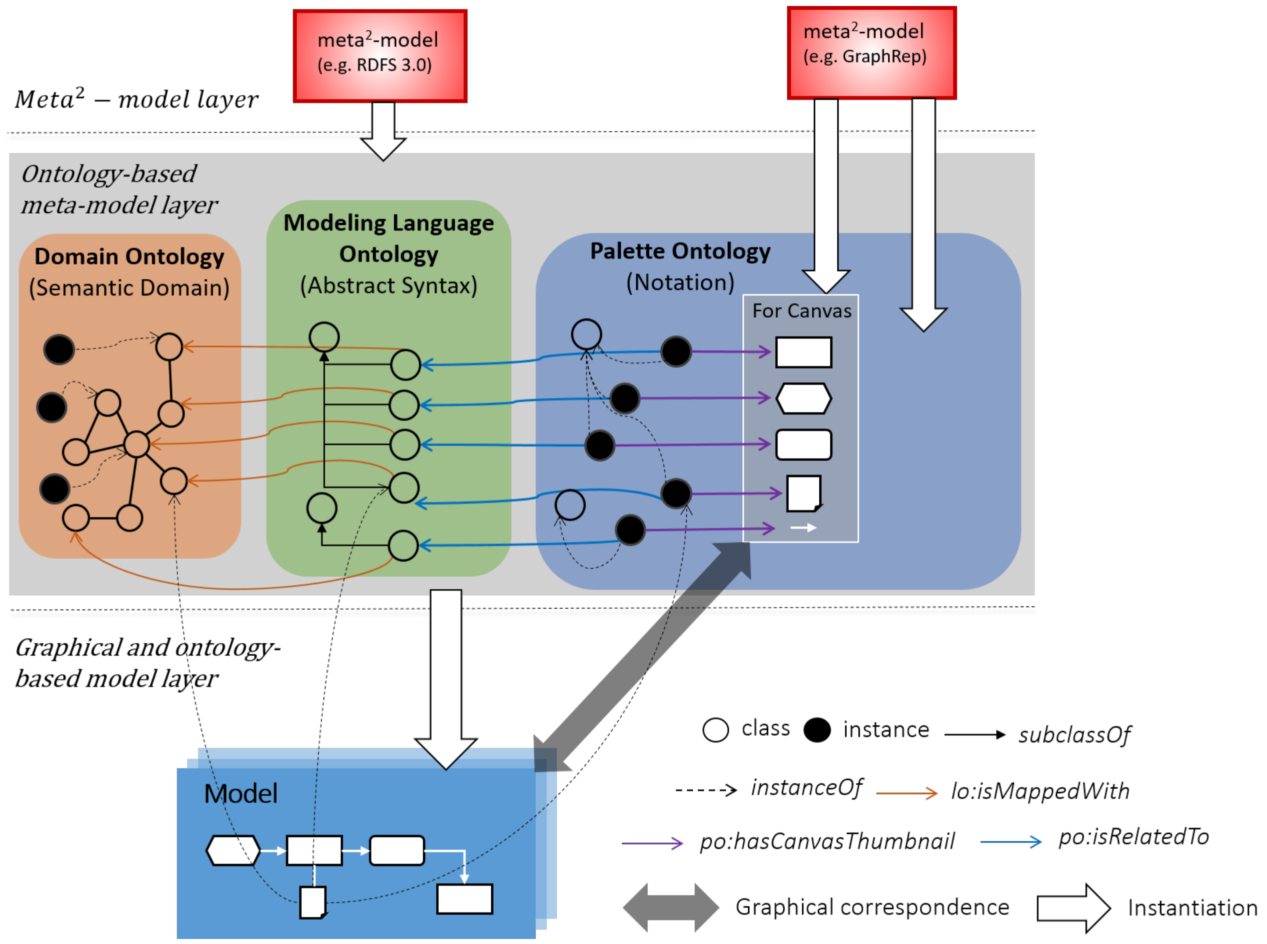
1. Engineering a baseline composed by a small set of Domain-Specific Semantic Process Model Pattern
   - Using AOAME
2. Engineering/Learning the initial similarity model following an OBCBR approach
   - Allows the retrieval of the semantic patterns matching with the process model
3. Adapt similarity model and learn new patterns
   - Domain Experts' Feedback

Phase 1 – Engineering Domain-Specific Semantic Process Model Pattern

AOAME - Seamless alignment between the human and machine-interpretable representation

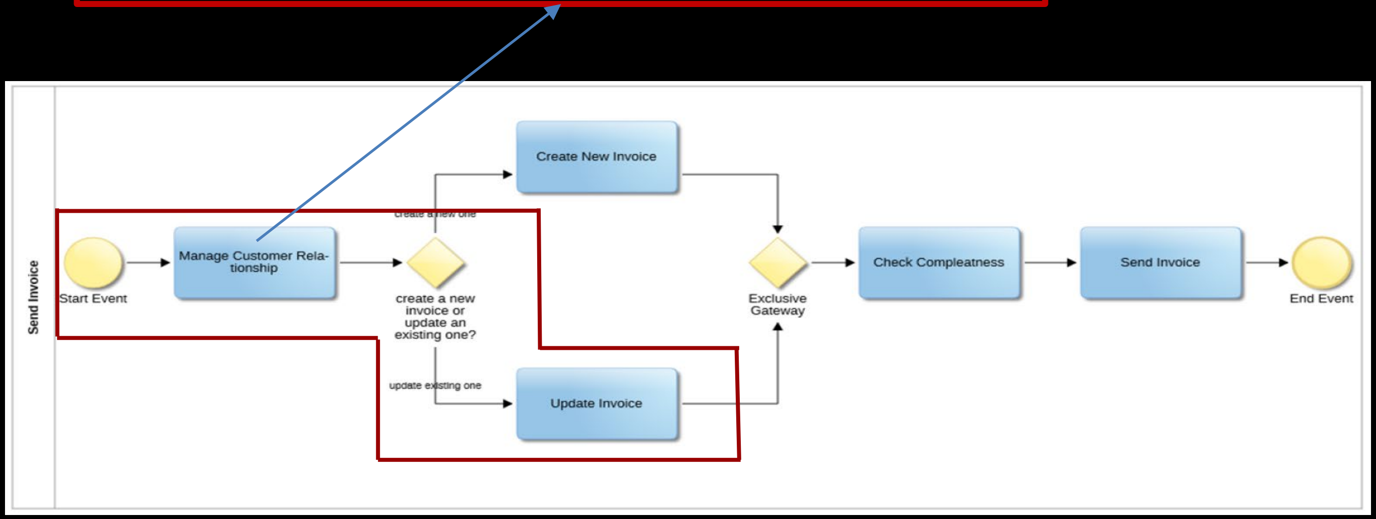


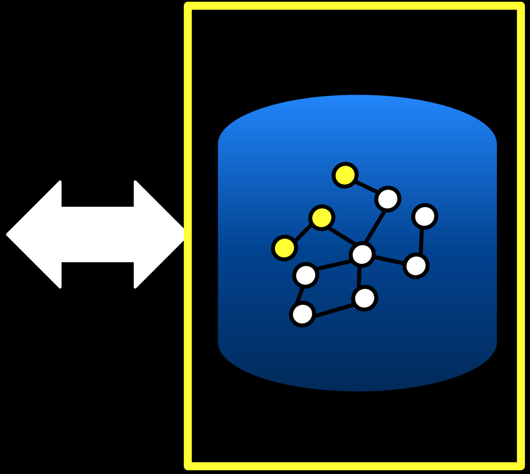
Ontology-based Metamodel


meta²-model (e.g. RDFS 3.0)
meta²-model (e.g. GraphRep)
Meta² – model layer
Ontology-based meta-model layer
Domain Ontology (Semantic Domain)
Modeling Language Ontology (Abstract Syntax)
Palette Ontology (Notation)
For Canvas
Graphical and ontology-based model layer
Model
class
instance
subclassOf
instanceOf
lo:isMappedWith
po:hasCanvasThumbnail
po:isRelatedTo
Graphical correspondence
Instantiation

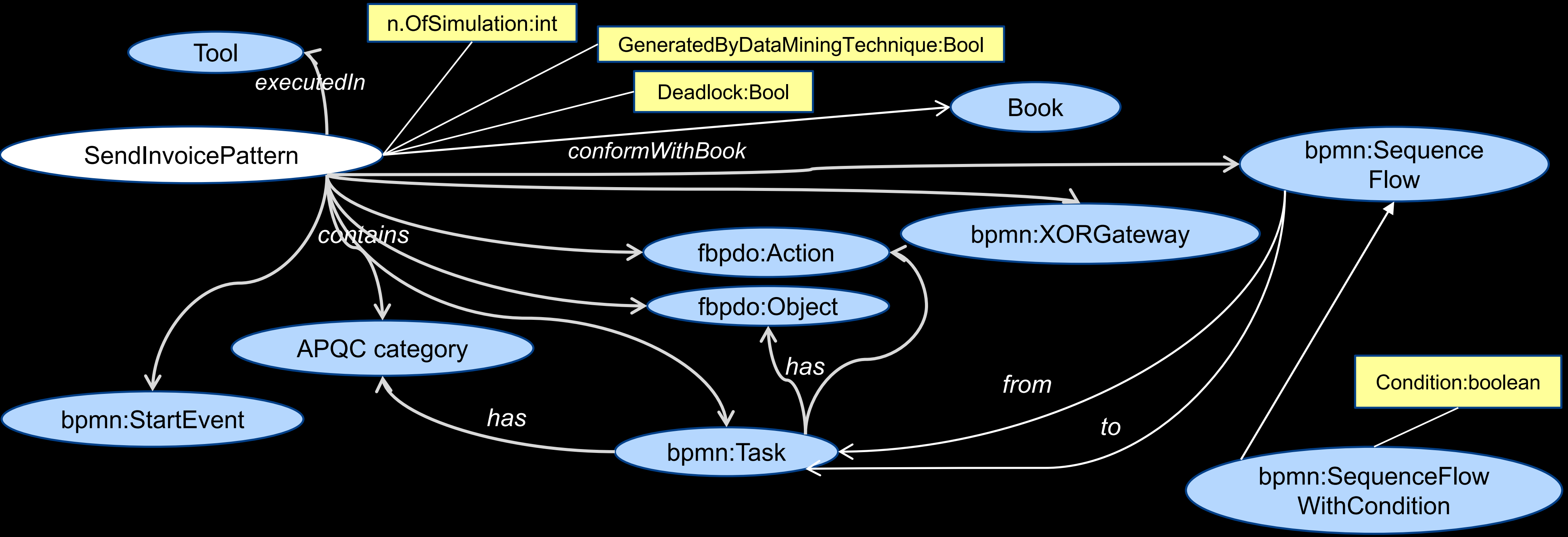
Send Invoice Pattern

Managing Customer Relationship
Action: Manage
Object: Customer
APQC: 3.5.2.4 Manage Customer Relationship

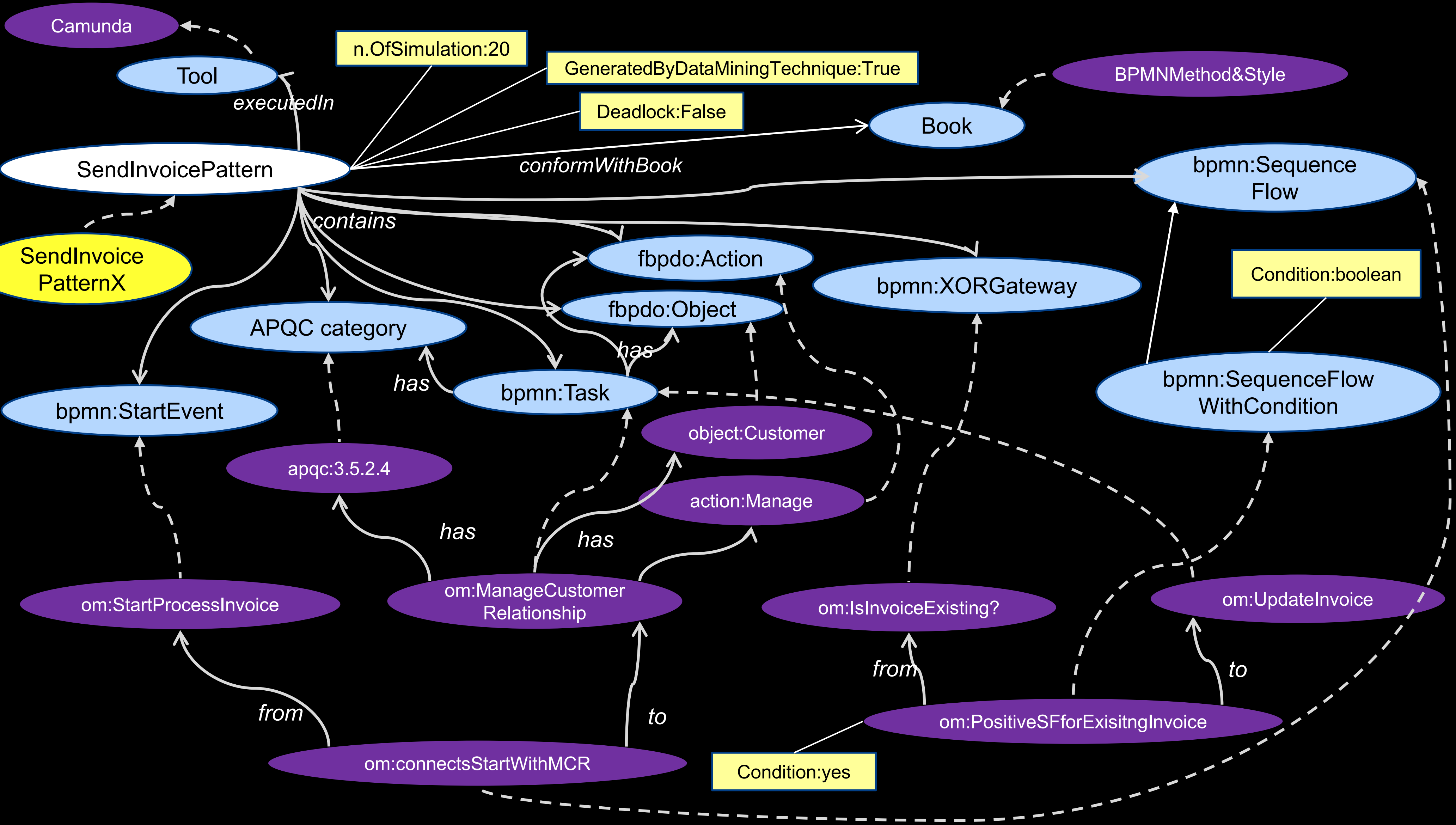




Domain-Specific Semantic Process Model Pattern (1/2)

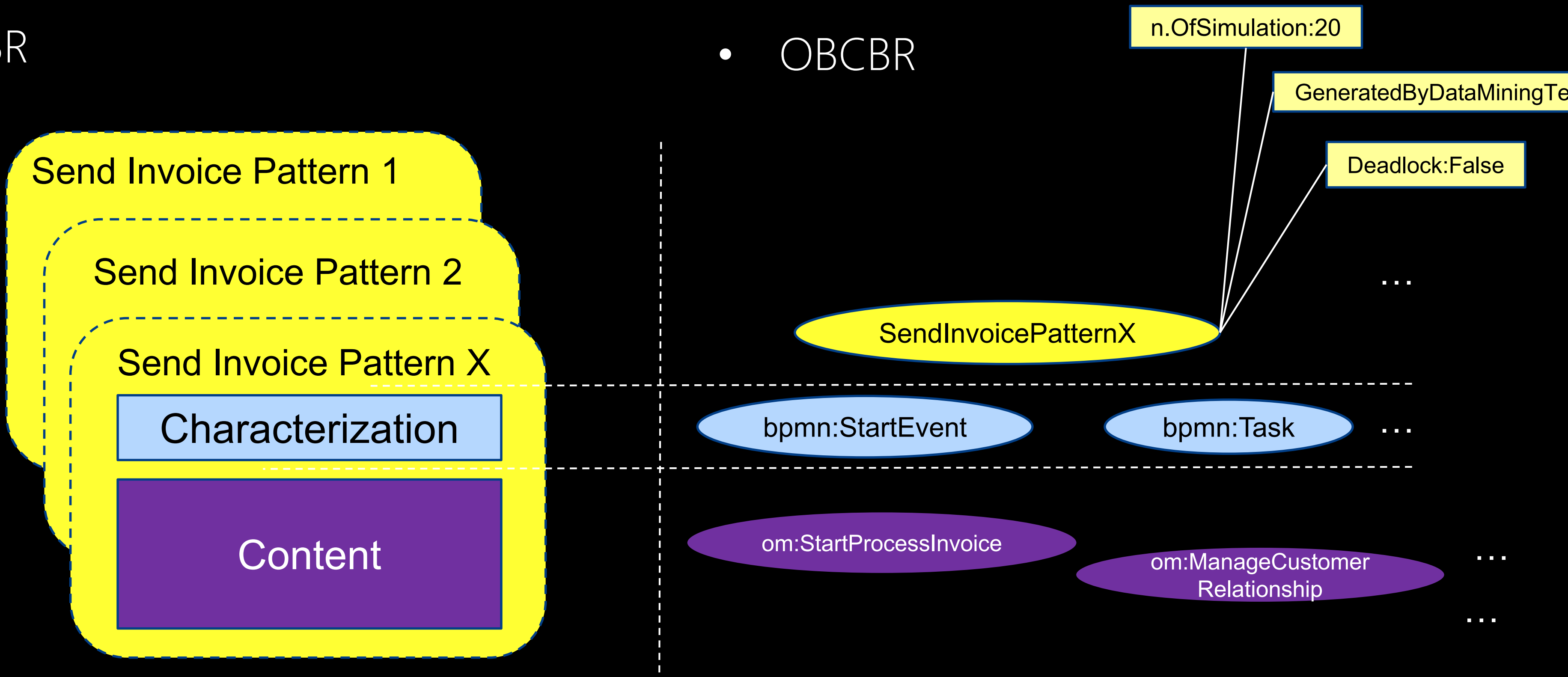


Domain-Specific Semantic Process Model Pattern (1/2)



Domain-Specific Semantic Process Model Pattern (2/2)

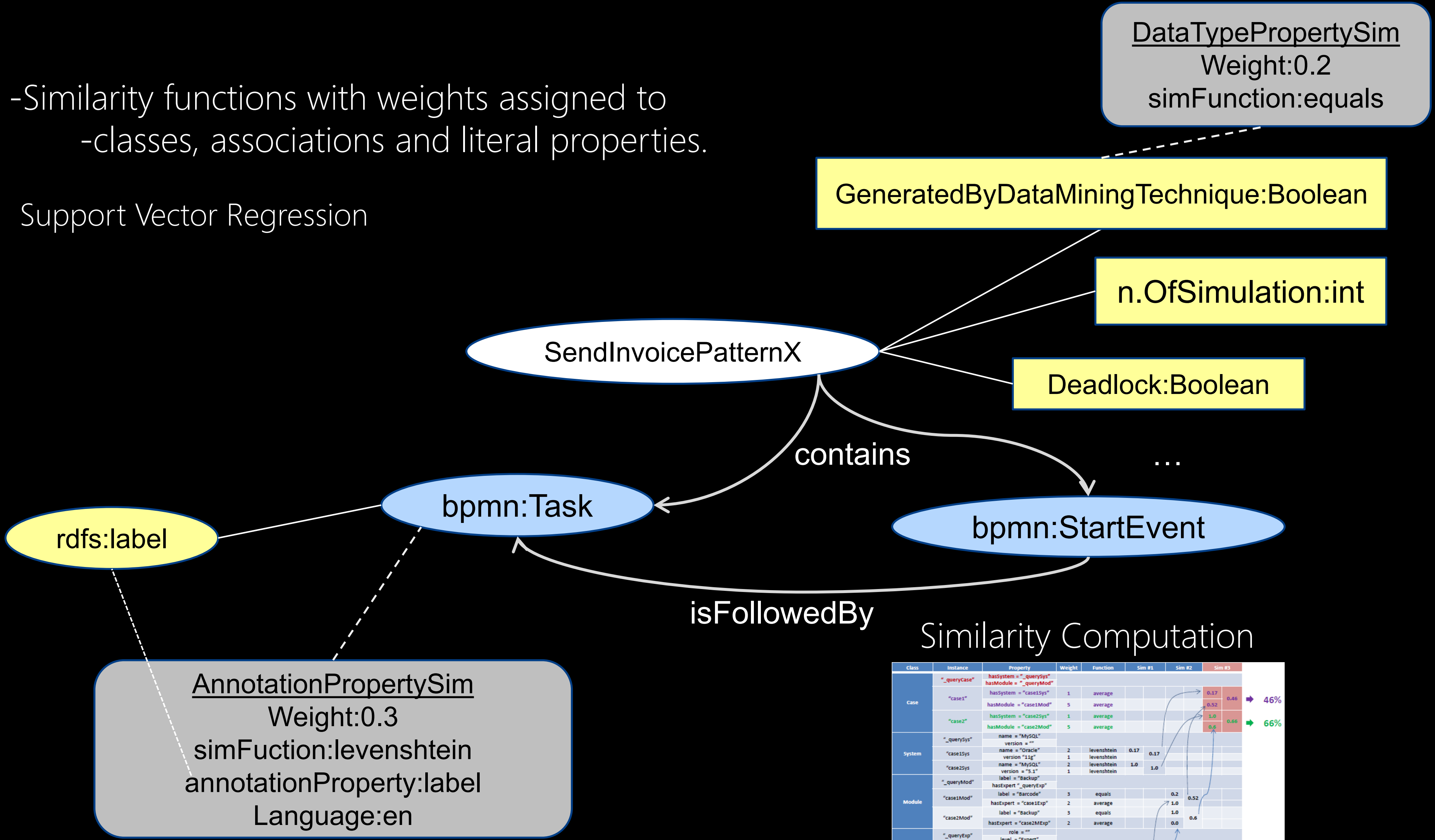
- CBR
- OBCBR



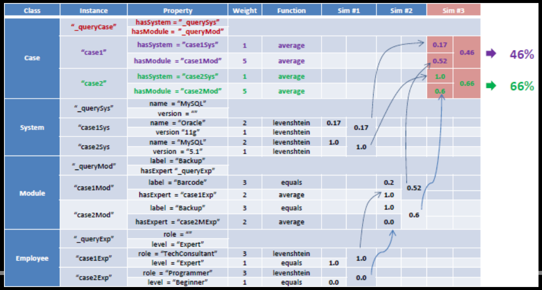
Phase 2: Engineering/Learning the initial similarity model

-Similarity functions with weights assigned to
 -classes, associations and literal properties.

Support Vector Regression



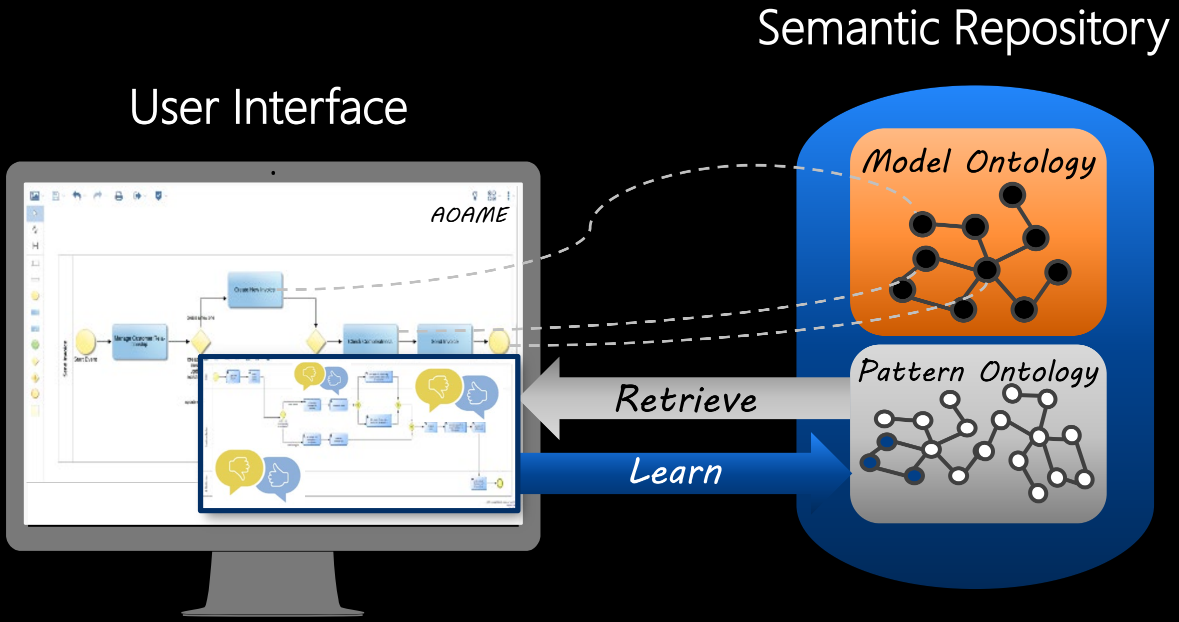
Similarity Computation



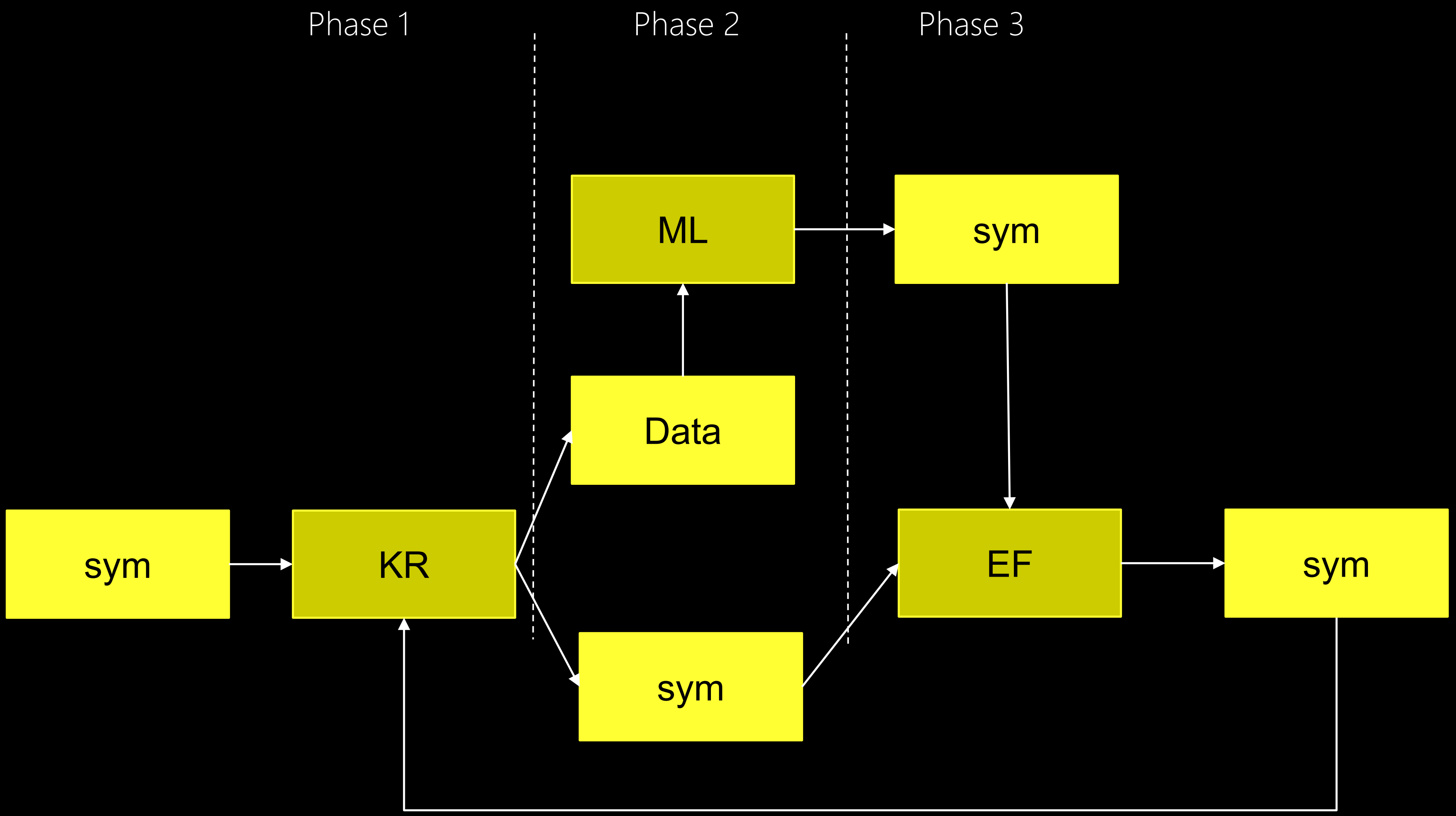
Class	Instance	Property	Weight	Function	Sim #1		Sim #2		Sim #3		
Case	"_queryCase"	hasSystem = "_querySys"									
		hasModule = "_queryMod"									
	"case1"	hasSystem = "case1Sys"	1	average					0.17	0.46	46%
		hasModule = "case1Mod"	5	average					0.52		
	"case2"	hasSystem = "case2Sys"	1	average					1.0	0.66	66%
		hasModule = "case2Mod"	5	average					0.6		
System	"_querySys"	name = "MySQL"									
		version = ""									
	"case1Sys	name = "Oracle"	2	levenshtein	0.17	0.17					
		version "11g"	1	levenshtein							
	"case2Sys	name = "MySQL"	2	levenshtein	1.0	1.0					
		version = "5.1"	1	levenshtein							
Module	"_queryMod"	label = "Backup"									
		hasExpert "_queryExp"									
	"case1Mod"	label = "Barcode"	3	equals			0.2	0.52			
		hasExpert = "case1Exp"	2	average			1.0				
	"case2Mod"	label = "Backup"	3	equals			1.0	0.6			
		hasExpert = "case2MExp"	2	average			0.0				
Employee	"_queryExp"	role = ""									
		level = "Expert"									
	"case1Exp"	role = "TechConsultant"	3	levenshtein		1.0					
		level = "Expert"	1	equals	1.0						
	"case2Exp"	role = "Programmer"	3	levenshtein		0.0					
		level = "Beginner"	1	equals	0.0						

Phase 3: Adapt similarity model and learn new patterns

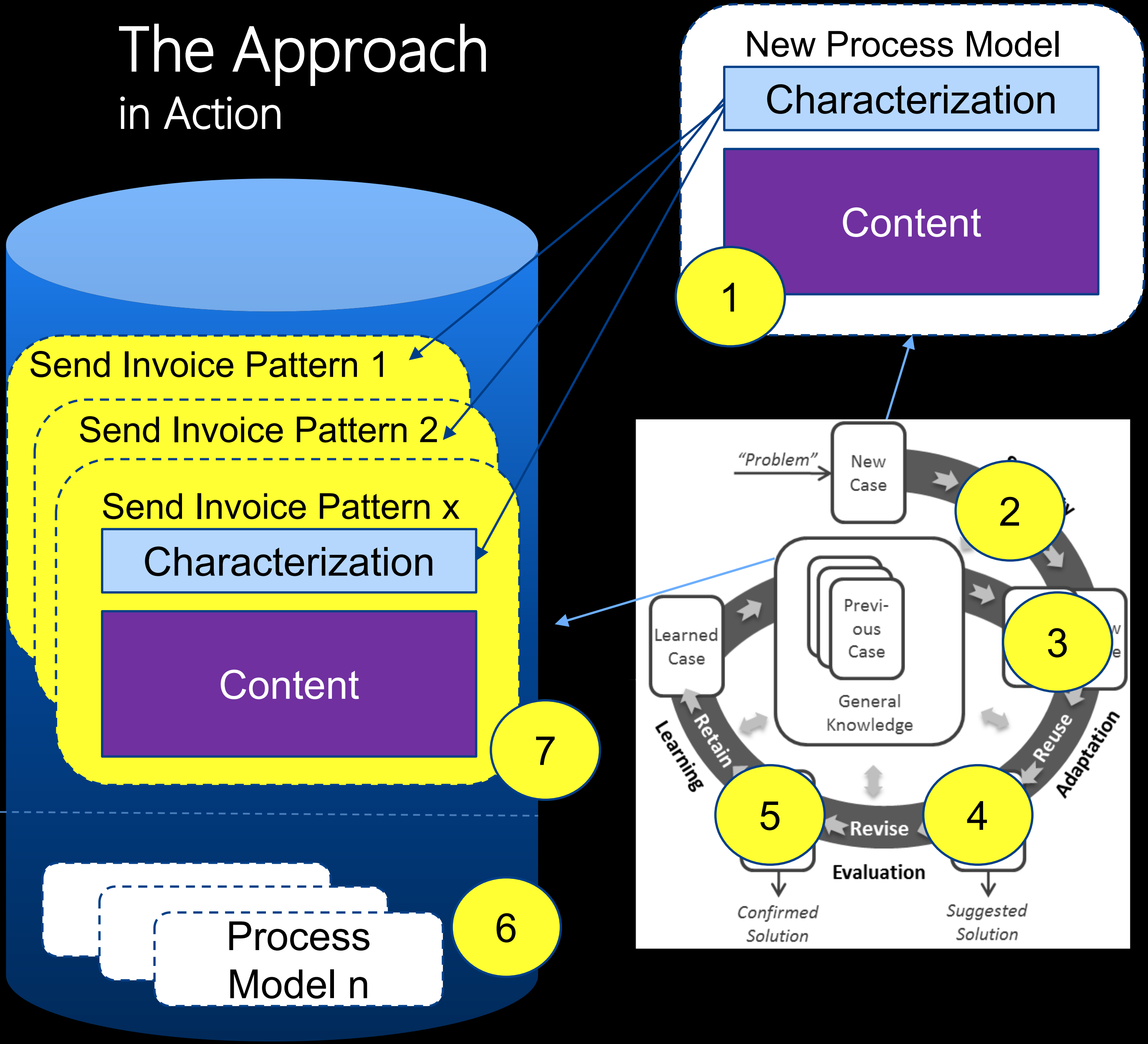
- Feedback on the retrieved Process Pattern Model
  - Aiming to improve the similarity model
- Decide on whether to add the revised model as a new Pattern in the Pattern Ontology



Into a Boxology



The Approach
in Action



1. Start New Process Model
2. Search fitting process model with case-based reasoning
3. Provide feedback on the retrieved patterns (re-make the similarity computation and repeat step 3 or go to 4)
4. Select and incorporate retrieved pattern to the current model
5. Revise the current process model
6. Retain the current process model in repository
7. Optionally add process as pattern

Thank you