

Expert Interview

First Part

When: 27/03/2018

Where: KnowGravity Inc., Hohlstrasse 534, CH-8048, Zürich

Expert: Markus Schacher

Role: Senior Consultant & Partner, Core Architect of KnowEnterprise

Company: KnowGravity

Website: <https://www.knowgravity.com/>

KnowEnterprise is a professional modeling environment that includes different modeling languages from GPMML like UML to more specific ones like BPMN (for process modeling), BMM (for business motivation modeling), OSM (for organization modeling) and SBVR (for rule modeling).

1. Did users ever have difficulties in learning standard modeling languages and using them in KnowEnterprise?

Certainly yes, but we try to avoid this. Usually we are directly involved in elaborating models either at the customer's side (as part of a mandate) or the customer participates in workshops on our side (in case of an inhouse project). This means that we carefully control the amount of learning needed by the customer and we always consider the skills the customer currently has or lacks. This means that we preferably introduce the modeling skills directly on a subject of the project and we extend them piece by piece. As example, we have just explained the idea of a "noun concept" and how to create them in KnowEnterprise to a railways domain expert and he was able to create a railways domain vocabulary. We didn't explain the whole SBVR specification before he was eligible to start working. Over time, he then learned more and more concepts of SBVR (and other language) as needed by the project and driven by his own curiosity.

- a. *If yes, what are the reasons for that?* Not applicable.
- b. *How could this issue be overcome?* See above.

In both research and industry, there is the recent trend of adapting standards (or existing) modeling languages to address a specific domain. In result a domain-specific modeling language is developed (DSML). This has the following benefits:

- It decreases the error prone while modeling. This is due the injection of semantics (i.e. abstract syntax and constraints) in the metamodel, which decreases the degree of freedom of modelers.
- It enhances understanding of models by domain experts. This is due to the graphical notations targeting a specific domain.

Developing a DSML through domain-specific adaptation of existing modeling languages has the following benefits:

- Modeling expert-friendly. This is due to the reference to already existing modeling standards.
- Total or partial reusability of the resulting language (i.e. DSML) within the modeling community. This is also due to the reference to already existing modeling standards.
- It fosters the quality of the modeling language. Established experience and lessons learned from existing modeling languages can be taken into account. Additionally, semantics and syntax can be borrowed.

2. Could a DSML address one of the issues/problems identified in question 1? Are there more problems that can be addressed? Not (directly) applicable. Since KnowEnterprise is based on a commercial UML modeling tool (PTC Integrity Modeler), all its supported

languages have been implemented as DSLs via "hard core" UML profiling (also called "Ergonomic Profiling" in the language of the tool provider).

3. *Have there been situations, where the modeling languages were not sufficient and where an adaptation could have made sense? Namely, adapting language constructs to fit a more specific domain?* In general, we try to avoid this. But KnowEnterprise supports various extension mechanisms:

- Model element types (e.g. BPMN Tasks) may be extended by adding additional properties (tags of their stereotypes) that may have declared as well as derived (i.e. computed) values
- Model element types (e.g. BPMN Tasks) may be specialized by specializing their stereotypes (e.g. «Compliance Task»)
- Model element types (e.g. BPMN Tasks) may be merged with aspects of different viewpoints (e.g.) by means of KnowEnterprise's built-in Aspect Weaver (e.g. a task might also be seen as a responsibility with some priority, enforcement, quality, etc.)
- Model element types (e.g. BPMN Tasks) may be extended by additional user functionality (e.g. a very specific consistency check)

Beside these possibilities, KnowEnterprise also supports adding entirely new (sub-)languages by means of loading additional (sub-)profiles via so called "KnowEnterprise Extensions". As an example, we created such an extension to support Value Flow Modeling (VFM) (part of the OMG specification "VDML", see www.omg.org/spec/VDML). VFM was not part of the core KnowEnterprise modeling languages but has been integrated with the organizational elements from its core (Organization, Organizational Unit, Position, Role, Person) that are able to produce or consume value.

- a. *If not, Why?* **Not applicable.**
- b. *Could a functionality for an on-the-fly customization of modeling constructs be useful in projects with domain-specific target?* **Yes, see above.**
4. *For the situations, where the language was not adequate, what kind of modifications on the modeling language would have helped? e.g. creating/deleting/update a modeling construct (i.e. class, attribute and relation on the metamodel level). (See answer to question 3 above)* In addition, to simplify user's life KnowEnterprise (and its underlying PTC Integrity Modeler) supports role-specific model browsers and role-specific subsets of menus (i.e. functionalities).
5. *Could you provide at least a use case where domain-modeling adaptation would have made sense?* As mentioned above, we have implemented KnowEnterprise Extensions for Value Flow Modeling, Safety and Risk Analysis (RIAL, including FMEA, FTA, ETA, HAZOP), extended responsibility modeling (RACI with rules), UML Testing Profile 2 (KnowEnterprise currently only supports UML Testing Profile 1), extended Requirements analysis and Evaluation, SEMAT (Software Engineering Method and Theory), Jackson's Problem Frame Approach (PFA), GUI/Web modeling, meta modeling, component library modeling, and more. Furthermore, we have also developed some domain-specific extensions to KnowEnterprise, particularly for the railways domain.

Second Part

1. *Do the operators derived in the paper (see below) make sense for you? I'm not sure whether I fully understand the intention of those operators (or better: "operations"?). These operators may be applied on the model-level as well as on the metamodel-level (i.e. the profile in KnowEnterprise). However (although possible in KnowEnterprise) I believe that the latter should not be used by the ordinary modeler/user. The possibility to modify the meta-model/ontology of a modeling tool by the "ordinary" modeler is particularly dangerous (and should be at least restricted) when already (important) model data exists: changing the meta-model could seriously damage the (potentially very valuable) model data.*
2. *Do you suggest other operators/actions on the language for adaptation purpose? Know-Enterprise (or better underlying Integrity Modeler) also supports renaming properties/relationships, refactoring of inheritance hierarchy, changing multiplicity and other constrains, changing property types, changing icons, symbols and visibility preferences of model element types, adding/updating/removing end-user functionality (context menus, toolbars, reports, events on model elements, etc.)*

Operators:

Operator 1: Create sub-class. This operator is applied on modeling elements and modeling relations to create new modeling elements and new modeling relations. This operator is also applied to integrate modeling elements (classes) from different modeling languages. For example, the operator would allow to connect "Discretionary Task" from CMMN as a subclass of the "User Task" from BPMN.

Operator 2: Delete sub-class. This operator is applied on modeling elements and modeling relations to remove unneeded modeling elements and modeling relations from the modeling language.

Operator 3: Create relation (object properties). This operator connects modeling elements and modeling relations to the related Domain Ontology concept.

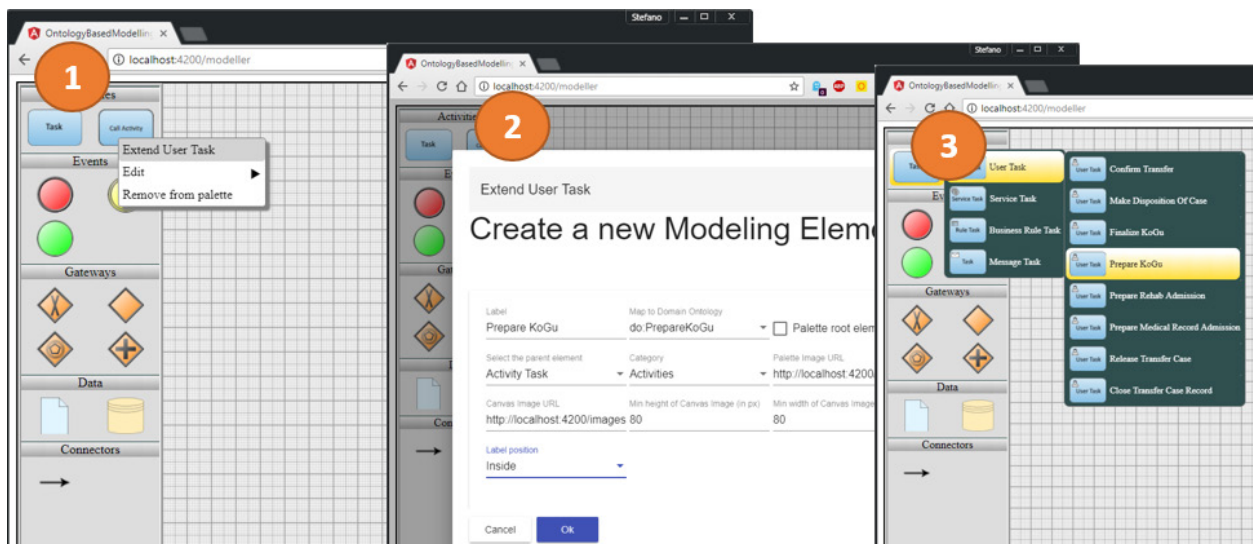
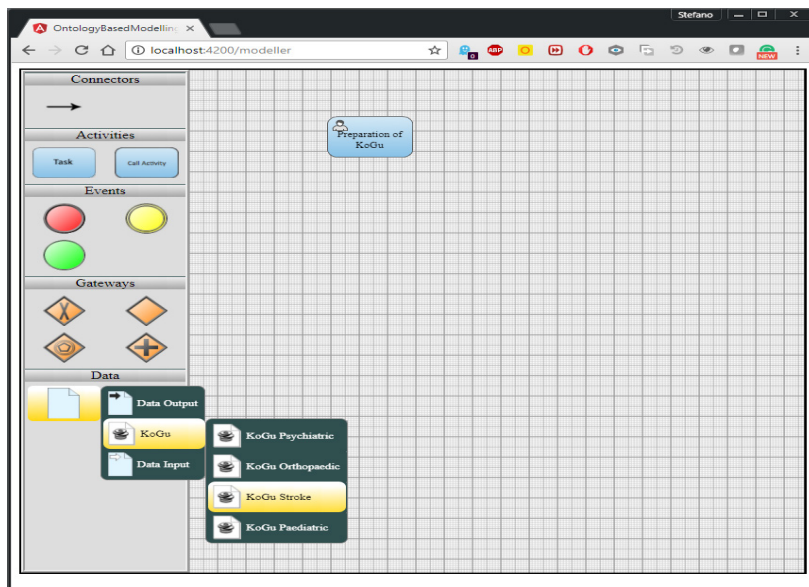


Figure 2. Operator from 1 to 3

Figure 1. Operator 1 applied on "Data Object"



Operator 4: Update relation (object properties). This operator is applied on as it allows updating existing connections between modeling elements/relations and the related Domain Ontology concepts.

Operator 5: Delete relation (object properties). This operator allows deleting existing connections between modeling elements/relations and the related Domain Ontology concepts.

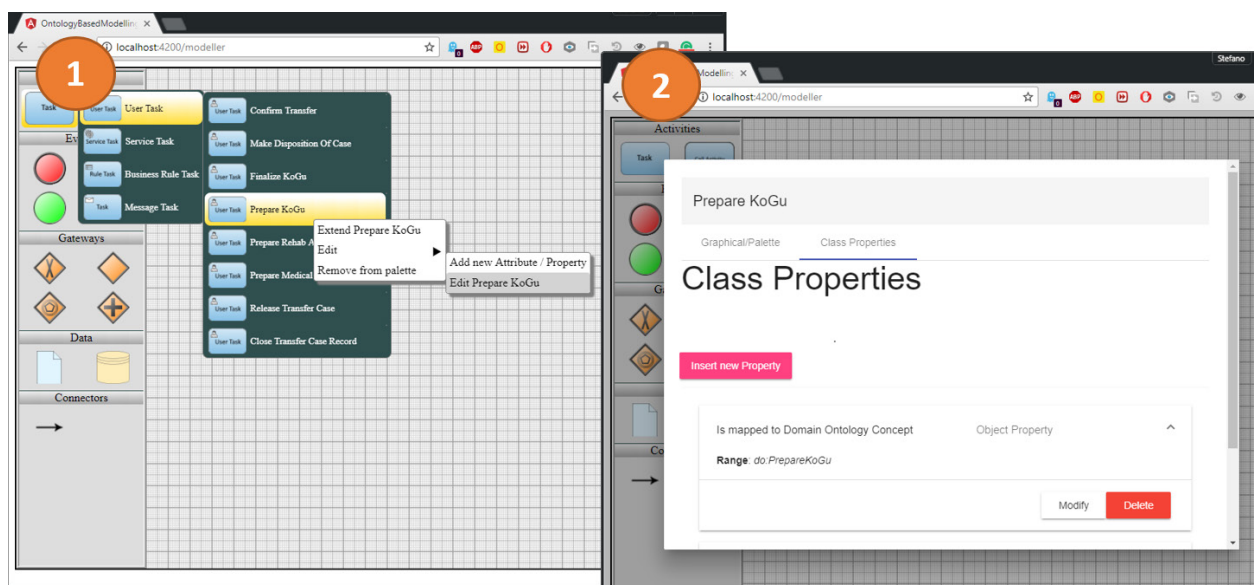
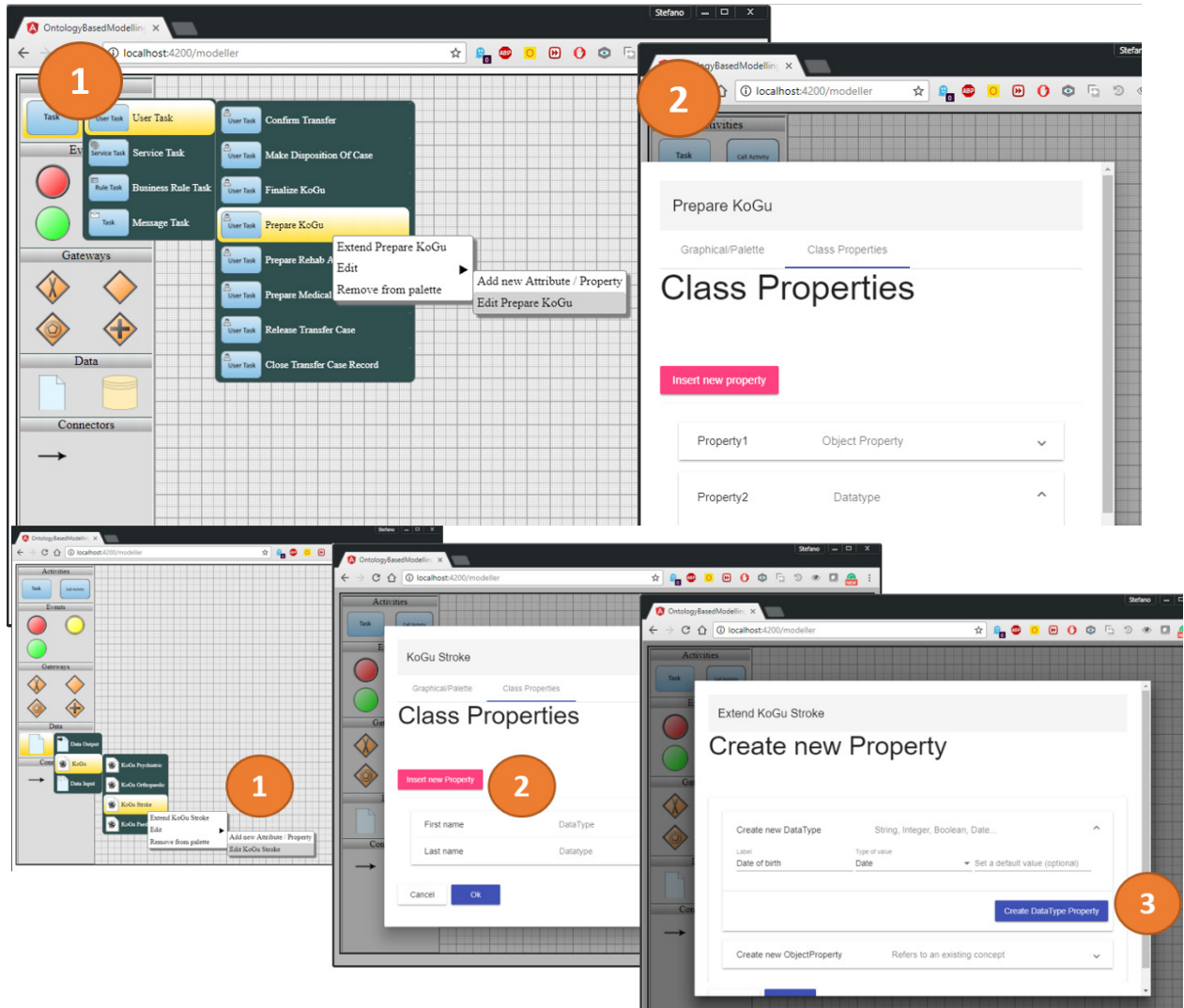


Figure 3. Operators 4 and 5

Operator 6: Create attribute (datatype properties). This operator allows adding new attributes to modeling elements and modeling relations.

Operator 7: Update attribute (datatype properties). This operator is allows updating existing attributes.

Operator 8: Delete attribute (datatype properties). This operator is allows deleting existing attributes.



Operator 9: Assign attribute type. This operator allows assigning value types String, Integer, Boolean to attributes of modeling elements.

Operator 10: Update attribute types. This operator allows updating types that are assigned to attributes of modeling elements.

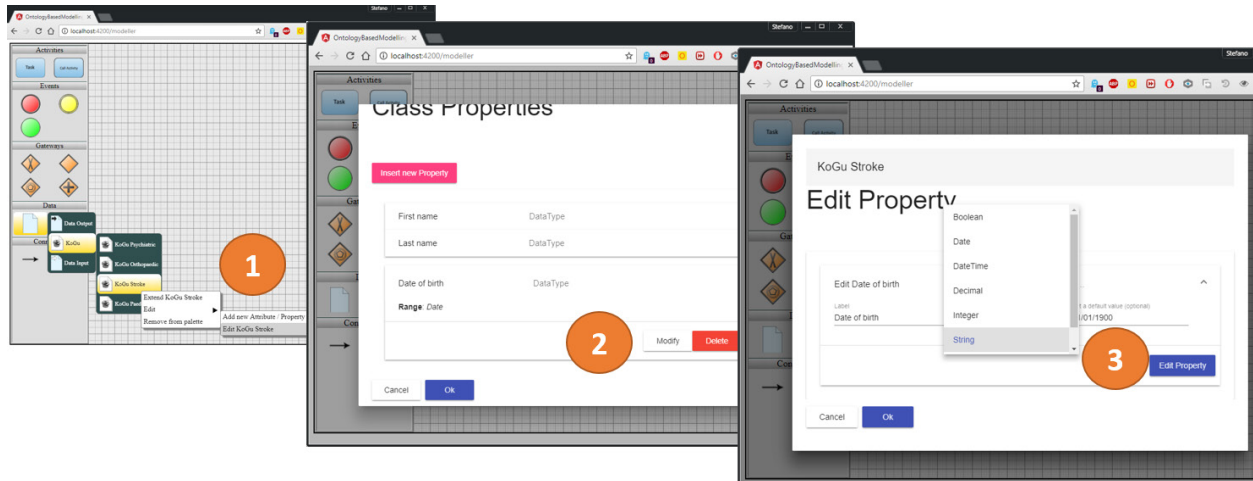


Figure 6. Operator 9 and 10

Operator 11: Enable representation level. This operator allows specifying whether a modeling element is a type or instance. Hence, the modeling environment can model instances as well as classes. Since an class can be an instance of another class, the modeling environment enables to distinguish between different levels of abstractions and not restricted to the class-instance dichotomy of description logics representation.

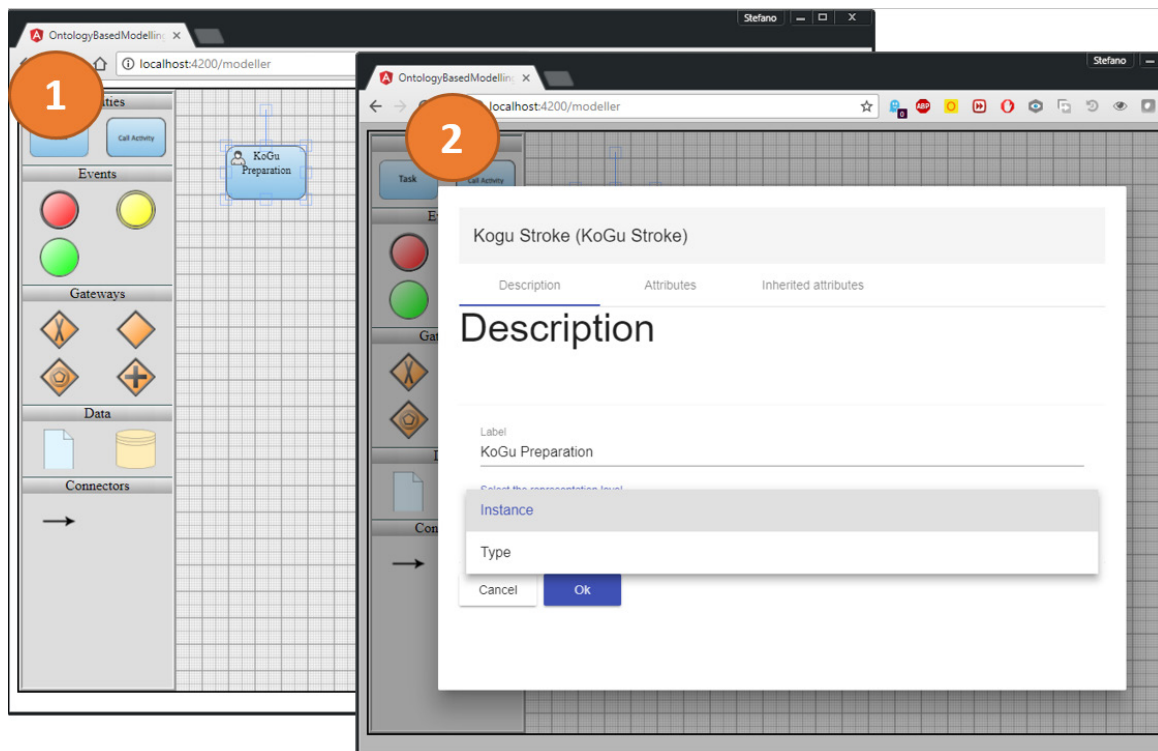


Figure 7. Operator 11

Figure 8 shows an excerpt of the ontology that is behind the modeling environment. It includes elements that belong to the architecture of the ontology-aided modeling environment, i.e. all elements with the “lo” prefix. Elements with the prefix “bpmn” belong to the modeling standard BPMN whereas “dslm4ptm” belong to elements of the created DSML that covers the patient transferal management domain. Finally, the element with the prefix “do” reflects the root concept of the domain ontology, which provides the (language-independent) semantics to both the modeling elements and modeling relations.

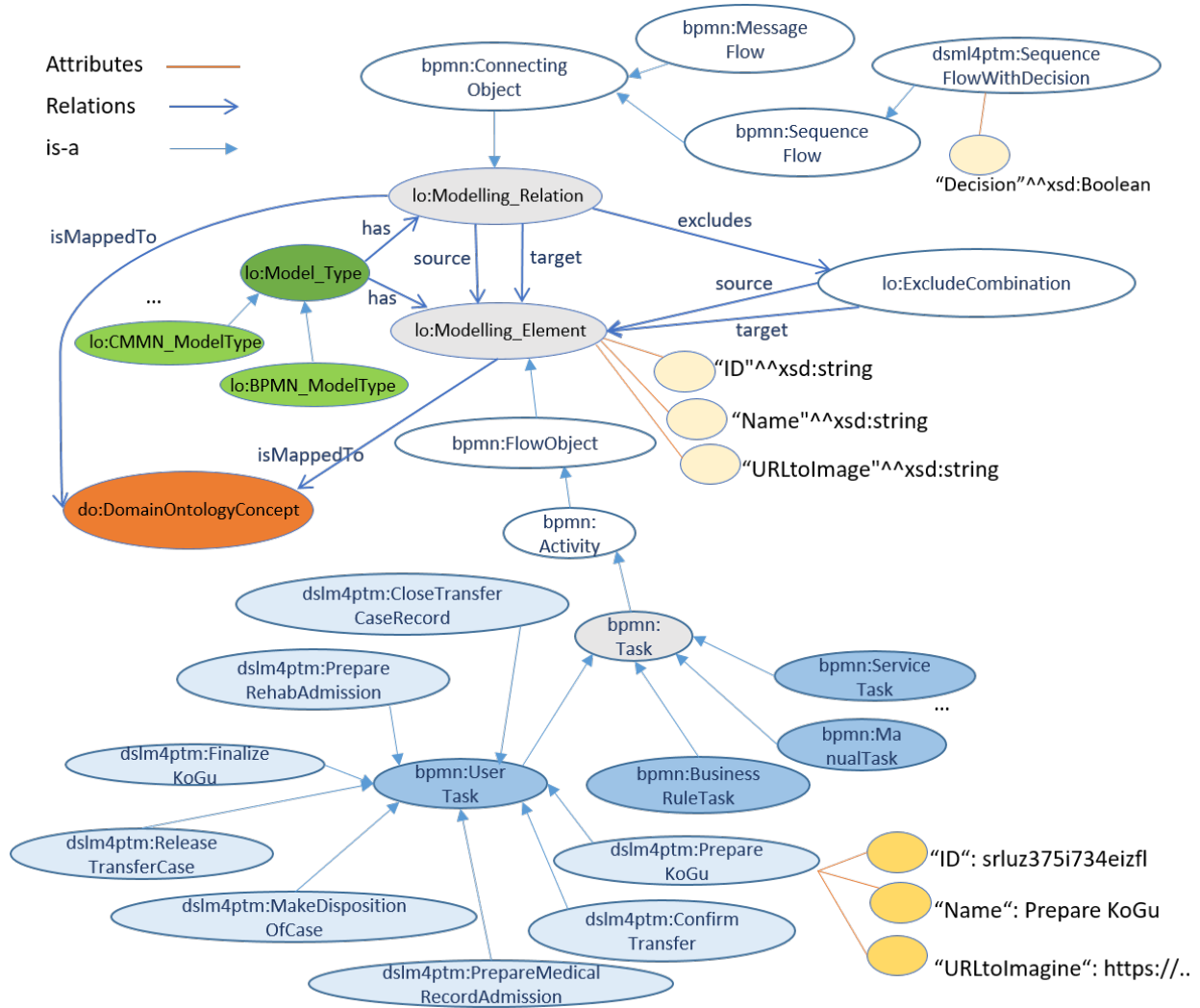


Figure 8. Ontology Excerpt

Figure 9 shows the the overall new approach agile and ontology-aided approach.

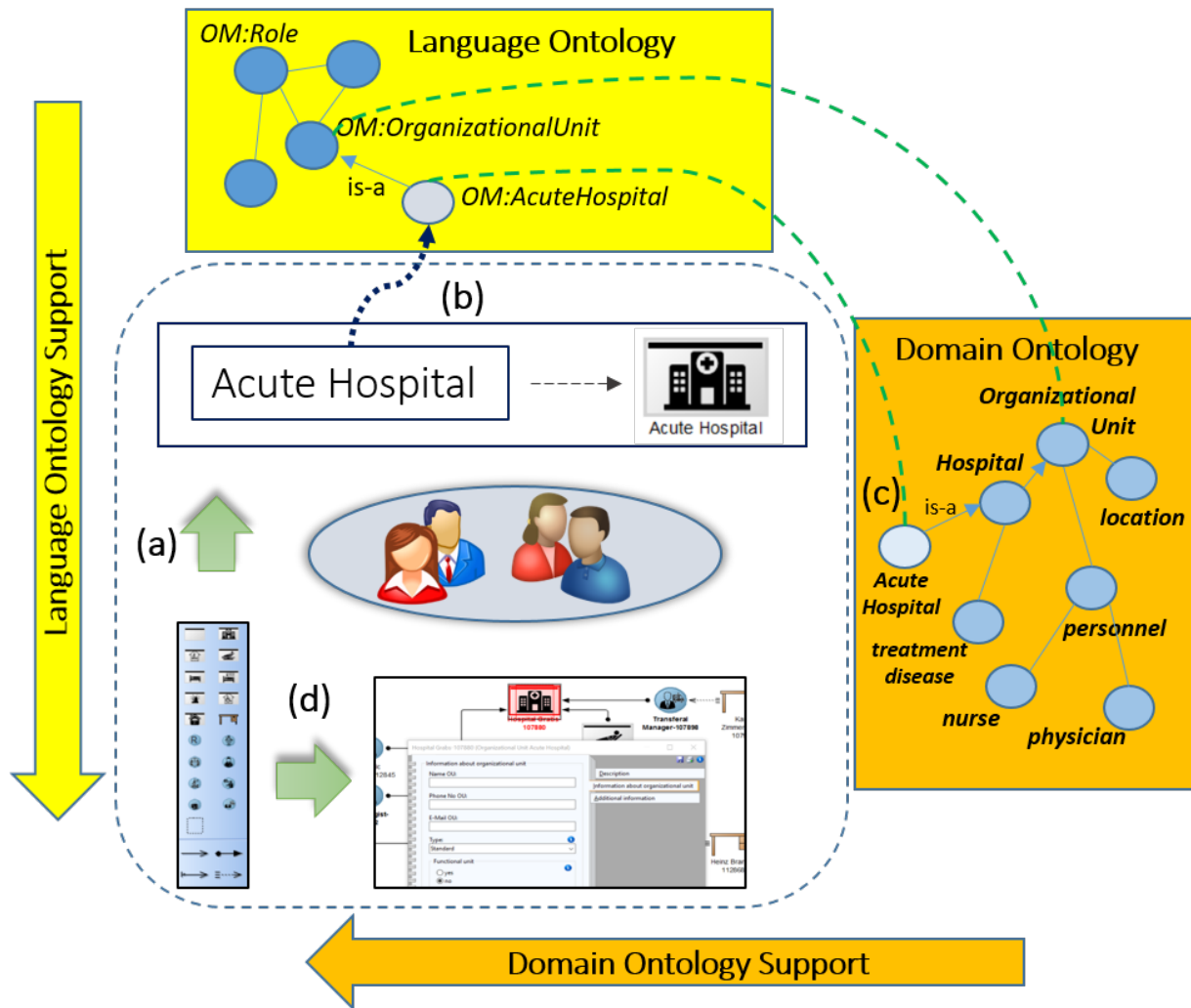


Figure 9. Approach