Ontology-driven Enterprise Modeling: A Plugin for the Protégé Platform

Benedikt Reitemeyer[1] and Hans-Georg Fill[2]

[1]Nuremberg, Germany
benedikt.reitemeyer@posteo.de
[2]Department of Informatics - Digitalization and Information Systems Group, University of Fribourg, Fribourg, Switzerland
hans-georg.fill@@unifr.ch

Abstract. The use of ontologies for enterprise modeling has been discussed from different perspectives in the past. In the paper at hand we describe design options for creating enterprise models by using an ontology as a shared domain conceptualization connected through ontology-driven conceptual modeling. The enterprise models thus act as representations of ontology instances. As a major benefit, a coupling between the visual representations of enterprise models and the reasoning capabilities that are typically available for ontologies can be achieved. In addition, we describe options for the realization of such an approach, which ideally builds upon existing platforms for enabling re-use and interoperability. Finally, we present an open-source implementation as Protégé plugin to show the technical feasibility and its application to a use case in the enterprise modeling area.

Keywords: Enterprise Modeling · Ontologies · System Design.

1 Introduction

Today, enterprises are confronted with large amounts of data, information, and knowledge that they need to process and manage [33, 38]. This concerns for example the analysis of data generated by sensors and engines in production environments, the interpretation of information from various enterprise resource planning systems or the representation of knowledge about business processes and organizational structures [30]. In order to support human actors confronted with these challenges, it is aimed for IT-based solutions that permit the representation, analysis and interpretation of these entities by machines. In this context, techniques developed in conceptual enterprise modeling, Enterprise Ontologies (EO) as well as in information visualization have been found to be beneficial [9, 17, 37]. Whereas conceptual models (CM) have traditionally focused on improving human understanding and communication by providing semi-formal, visual modeling languages, ontologies in the sense of formal knowledge representation are directed towards machine processing and automated reasoning [36, 41]. In addition, techniques in information visualization are used to amplify the cognition

of information and knowledge [9]. As interface between conceptual modeling and ontologies, the concept of ontology-driven conceptual modeling (ODCM) has evolved aiming on the use of ontological theories for the improvement of conceptual modeling [21]. Although CMs and visualizations share the aspect of graphical representations, they fundamentally differ on the level of semantics. Due to the underlying schema in the form of a modeling language, CMs may be interpreted by machines, at least to a certain extent [5]. In contrast, information visualizations are directed towards human interpretation only and their elements typically have no formal meaning assigned. An exception are visualizations of ontologies, e.g. as available via plugins for ontology editors such as Protégé. Furthermore, information visualizations are generated through algorithms with no or limited editing capabilities for the underlying information structures, whereas CMs may either be created and edited by humans or generated automatically [36, 38]. In summary, CMs have the advantage of a visual representation and analysis of knowledge and information structures, ontologies enable machine processing and reasoning based on formal axioms and information visualization targets the analysis of data by amplifying cognition. It thus seems beneficial to investigate ways of how the benefits of these three directions can be joined. This applies both to the design of according solutions as well as their technical realization. In the following we will thus propose design options for what we will call *ontology-based enterprise modeling* and outline how these options can be implemented.

The remainder of the paper is structured as follows. In section 2 some fundamental terms will be defined to achieve a common understanding. Section 3 shows possible technical realization options. Subsequently, an example implementation of a realization option set is shown in section 4. In section 5 a practical use case of the approach is discussed. Benefits and Drawbacks of our approach are evaluated in section 6. The paper ends with a brief outlook and conclusion in section 7.

2 Foundations

In this section we will briefly explain some fundamental terms in regard to CMs, visualizations, and ontologies in order to achieve a common foundation for the following elaborations.

2.1 Conceptual Models and Visualizations

CMs and visualizations differ in several aspects, which shall be outlined according to the dimensions *syntax*, *identity*, *semantics*, and *machine interpretability* - see also Table 1. First, CMs - in the way we regard them here - are language-based. They depend on a system of symbols and rules for the combination of those symbols, i.e. a grammar or syntax [40, 24]. For CMs, the syntax is typically defined in a formal language to ensure its exact interpretation [5]. In contrast, the syntax of visualizations depends on their implementation. If a visualization is constructed using vector graphics, at least the graphical representation and

the rules for its composition are based on a formal specification, which can be viewed as a kind of syntax. Otherwise, when pixel graphics are used for their representation, there exists no formal structure at all [12].

Furthermore, elements and relations in CMs have an identity. They can always be distinguished from other elements based on unique identifiers, which permit to modify them individually. Elements in information visualizations do not need to be uniquely identified. Although they may be associated to certain data values that may or may not be unique, they typically need not be individually accessible and modifiable.

While the syntax defines fundamental structures and the way these may be created, its major purpose is to assign semantics, i.e. meaning, to these structures. Here, a large difference between CMs and visualizations exists. Whereas CMs are based on schemata that carry meaning a-priori - either formally defined or given in natural language - information visualizations are directed towards human interpretation, where meaning is assigned to graphical elements as needed, i.e. ex-post. This leads to implications for machine processing. When constructing algorithms for interpreting content, it is considerably easier to do so for CMs due to their grammar and fixed meaning than for information visualizations [5].

	Conceptual Models	**Visualizations**
Syntax	Formal	Formal in case of vector graphics; None in case of pixel graphics
Formal Semantics	Optional	None
Identity of elements and relations	Yes	No
Machine interpretability	Yes, depending on the level of formality	Typically not, only directed towards humans

Table 1. Differences between Conceptual Models and Visualizations.

2.2 Ontology-Driven Conceptual Modeling

Ontology-driven conceptual modeling can be explained as a conceptual connection between CMLs (CML) and ontologies. Whereas different kinds of CMLs and ontologies are combined, ODCM approaches can be characterized by different kinds of phenomena: *static phenomena*, *dynamic phenomena*, and *behavioral and functional phenomena* [44].

In a common definition, ODCM is described as the application of ontological theories, based on broader ontological areas as formal ontology, cognitive science or philosophical logics. Those theories are practically applied in areas such as

the development of engineering artifacts, improving the theory, or conceptual modeling [21]. In concrete, various ontology-driven CMs based on different ontology types have been developed, e.g. in the area of foundational ontologies, business and EOs, database design and architecture, or software systems development and architecture [44]. One of the most used foundational ontologies is the Bunge-Wand-Weber (BWW) ontology which was developed to illustrate an information system and provides definitions of important concepts like system, subsystem, and coupling [45]. While the BWW ontology is very high-level, there are more concrete ODCM approaches like the Unified Foundational Ontology (UFO) which has been constructed, aiming on both, improvement of conceptual modeling as theoretically discipline and improvement of practical implications and is used as basis for e.g. OntoUML [3, 23] or the UEML approach that puts the BWW into practical use [37]. UFO itself instead aims on improving the semantics of CMLs and not on improving the semantics of specific models.

In a meta study, Verdonck and Gailly [44] count the frequency of scientific appearances of different ontologies and CMLs in ODCM and classify them in terms of their perspective. They distinguish between static, dynamic, and behavioral and functional perspective. Phenomena in the static perspective describe the structure of a system, such as entity, thing or objects. In contrast, phenomena in the dynamic perspective represent change and time. Lastly, phenomena in the behavioral and functional perspective are social phenomena and states with their transitions.

2.3 Conceptual Enterprise Models and Enterprise Ontologies

CMs and information visualization can represent enterprise knowledge and information mainly directed to human processing. In contrast, ontologies, as formal representations of knowledge, offer machine processability. Therefore, the differences between CMs and ontologies especially in the enterprise knowledge context must be considered. For our distinction of conceptual enterprise models and EOs the dimensions *purpose and goals*, *formal foundation*, *adequacy for automated reasoning*, *inherent visual representation*, and *human-adequate structuring* are considered (see Table 2).

Conceptual enterprise modeling concerns the creation of integrated enterprise models and sub-models which aim at capturing enterprise aspects required for the modeling purpose. Aspects captured by CMs are for instance processes, business rules, or concepts (e.g. information, vision, goals, and actors). Based on those aspects, current and future states of the enterprise are described. Additionally, the conceptual enterprise models contain the enterprise knowledge of the stakeholders which are involved in the modeling process [7]. Conceptual enterprise modeling methods in our understanding are for example the Business Modeling and Notation (BPMN), ArchiMate, or Multi Perspective Enterprise Modelling (MEMO) [18, 28].

Ontologies are characterized as "a shared and common understanding of some domain that can be communicated across people and computers" [41, p. 186]. They are represented formally and require capabilities of underlying formal

axioms for reasoning and inferences to detect new knowledge [20]. To measure the quality of an ontology various criteria are suggest, e.g. clarity, consistency, accuracy or applicability [8, 19, 42]. The ontology quality can also impact what can be achieved within the reasoning. In an enterprise context, ontologies have the purpose to acquire, represent, and manipulate knowledge based on a formal description of the deep structure behind the surface of an enterprise [11, 43]. Examples for EOs are TOVE (TOronto Virtual Enterprise) ontology [16] and the Unified Foundational Ontology (UFO) [22].

As we have already shown in section 2, CMs are formally language-based. Conceptual enterprise models, as a specialization of CMs, are therefore language-based as well. Beneath the language-base the meta meta models and meta models are used for structuring conceptual enterprises hierarchically. Meta models and meta meta models are created with own modeling languages, which describe the components of the underlying level. In this way, different abstraction levels of models are formalized.

The formal foundation of ontologies is based on formal languages as well. Various different languages like the Resource Description Framework (RDF), DAML+OIL or the Web Ontology Language (OWL) are used for defining the ontologies [1, 31, 34]. In this paper we will limit our focus to OWL, because of its widespread use in ontology engineering and design, and its standardization. OWLClasses are groups of individuals that belong together, because they share properties. Therefore, OWLProperties are needed to state the relationship between individuals or from individuals to data values. Individuals are instances of classes. Our third dimension for the differentiation of conceptual enterprise models and EOs is the adequacy for automated reasoning. Wang et al. [46] differ between ontological reasoning, which is based on a set of first-order formulas specified through description logic, and user-defined reasoning, which allows users to define their own reasoning rules, e.g. the Semantic Web Rule Language (SWRL) [27]. Therefore, EOs, due to their use of description logics, are adequate for automated reasoning, while conceptual enterprise models are typically not.

For conceptual enterprise models an inherent visual representation is defined by the graphical notation, as well as the ordering rules in terms of the syntax. They aim at integrating multiple perspectives or views to create a detailed and complete description of the enterprise [4] In contrast, for EOs those inherent visual representations don't exist. This difference is based on the former explained purposes. As EOs are directed towards machine processing, they don't need an inherent visual representation or human-adequate structure. Conceptual enterprise models are directed to human understanding and therefore are human-adequate structured, based on an inherent visual representation.

In enterprise context exist various approaches either for conceptual enterprise models, as well as for EOs. As the analysis of this section showed, both, conceptual enterprise models and EOs, aim for representing enterprise context. Conceptual enterprise models aim on the human-adequate construction of integrated enterprise models, based on an inherent visual representation. In contrast, EOs target the acquisition, representation and manipulation of enterprise

	Conceptual Enterprise Models	Enterprise Ontologies
Purpose and Goals	Integrated enterprise models and submodels for human understanding	Acquire, represent and manipulate machine processable knowledge
Formal Foundations	Semi-formal	Formal
Adequacy for Automated Reasoning	Typically not	Yes
Inherent Visual Representation	Yes	No
Human-adequate Structuring	Model/ Diagram Types	No

Table 2. Differences between Conceptual Enterprise Models and EOs.

knowledge, based on a formal foundation, which is machine-processable and appropriate for automated reasoning. Further, the occurring enterprise phenomena can be described from different perspectives, helping to close the knowledge gap between users and modelers. Trying to use the benefits of both concepts, in section 3 different realization options for ontology-based enterprise modeling are developed.

3 Possible Realization Options for a Technical Realization

Based on the presented foundational considerations several technical realizations options for an enterprise ODCM software application are possible. In this section those options are discussed, based on *ontology editors*, *enterprise modeling editors* and *hybrid editors*, which combine conceptual modeling and additional semantic information. Subsequently, the options are discussed, and a new approach is introduced. The research method used for this section is argumentative-deductive reasoning [48].

3.1 Ontology Editors

For classifying different ontology editors various characteristics, such as editing and browsing are used [47]. As the focus of our work is on *conceptual modeling* and *visualization*, the characterization of the ontology editors focus on those two dimensions and further on the ability of generating knowledge through *reasoning*, and the structure of the foundational *domain knowledge* and *modeling knowledge* - see also Table 3. The characterization includes the editors themselves and additionally possible extensions (e.g. plugins).

Several open-source and commercial ontology editors exist in science and practice. Well-known and widespread editors are for example *Protégé*[1], *TopBraid*

[1] https://protege.stanford.edu/

Composer[2], and *OntoStudio*[3]. The probably most known and used ontology editor is Protégé. Protégé is an open-source platform, offering various external developed plugins [35]. It offers extensive functionality, including visual representation of the developed ontologies, but is lacking the ability of visual modeling ontologies. The commercial ontology editor TopBraid Composer is based on the Eclipse platform and therefore offers many features and plugins as well [10]. As Protégé it offers ontology visualization functionalities and additionally the capability of modeling ontologies in an uml-like style. The third example for an ontology editor is OntoStudio which is as well based on the Eclipse platform [47]. OntoStudio and its plugins offer visualization functionality but miss the ability of modeling ontologies.

3.2 Enterprise Modeling Tools

Beneath investigating ontology editors in terms of their modeling functionality, enterprise modeling editors can be investigated in terms of their ontology use. A large number number of different enterprise modeling editors such as the MID Innovator or Sparx Systems Enterprise Architect are used. As far as we investigated, there are currently no full enterprise modeling tools which are based on the ODCM approach. For individual CMLs approaches and implementations for ODCM related editors can be found. For example, Benevides and Guizzardi [3] developed an editor for conceptual modeling and ontology engineering. While there are few ODCM editors, still several approaches for connecting CMs and ontologies are developed under the term *Semantic Lifting*.

3.3 Semantic Lifting and Semantic Annotation

Semantic lifting is defined as "the process of associating content items with suitable semantic objects as metadata to turn unstructured content items into semantic knowledge resources" [2]. Hinkelmann et al. [25] explain various ways for the application of semantic lifting on meta models referencing different research projects. A practical implementation of the semantic lifting approach is realized in the SeMFIS tool based on the ADOxx platform [13, 14]. SeMFIS is an editor for semantic annotations of CMs. It is based on various sets of meta models which enable the visual representation of ontologies and semantic annotations as models [15].

In conclusion, Ontology editors offer the ability to visual represent knowledge, but have only partly the ability for the creation of models. They offer a reasoning capability, enabling reasoning on the formal domain knowledge. While the modeling knowledge for the modeling ability is not represented as ontology. In contrast, modeling editors offer the ability of creating models and visual representations but have typically no capability for reasoning on the resulting

[2] https://www.topquadrant.com/tools/modeling-topbraid-composer-standard-edition/

[3] http://www.semafora-systems.com/en/products/ontostudio/

models. Lastly, hybrid editors, like SeMFIS, enable the visual representation of knowledge, modeling, and reasoning over the results. The domain knowledge is added with semantic lifting in a formal way.

	Ontology Editors	Modeling Editors	Hybrid Editors
Visualization	Yes	Yes	Yes
Modeling	Partly	Yes	Yes
Reasoning	Yes	No	Yes
Domain Knowledge	Formal	Informal	Formal
Modeling Knowledge	None	Informal	Informal

Table 3. Comparison of Characteristics of Ontology, Modeling and Hybrid Editors.

Relating to the targets described in section 2, creating knowledge that is human and machine-processable and closing the knowledge gap between users and modelers and between modelers we propose a new realization option combining the best of both, ontology editors and modeling editors. For the new approach, the domain knowledge and modeling knowledge are represented as ontologies and combined through mapping, thereby enabling the machine-processing with reasoning functions of ontology editors and creating a common knowledge base for users and modelers. This idea extends the approach of Hinkelmann et al. [26] of ontology-based meta modeling in which ontological meta models are extended with the graphical notations. In addition, a modeling function should be added for creation of CMs, making modeling and the processing of knowledge by human stakeholders possible. Altogether, the approach is the realization of the ODCM idea of combining ontologies and CMs.

After the theoretical description of the possible existing realization options and a new realization approach, section 4 introduces the technical realization of the approach, describing our approach as a Protégé-Plugin.

4 Realization as a Plugin of the Protégé Ontology Platform

In this section we describe the implementation of the previously introduced approach in terms of the *Ontological Foundation*, *Technical Foundation*, and *Modeling Mechanism* as a plugin for Protégé platform. The research method used for the development and evaluation of the plugin was prototyping [48].

The approach is based on an ontological foundation consisting of a domain ontology illustrating the technical domain of an enterprise, department, or mar-

ket sector and a modeling ontology containing the entities and relations of a CML and their necessary attributes like the graphical representation. As it is a widespread ontology language, both ontologies are implemented in OWL. The conceptual idea is to offer an approach which is as generic as possible to be able to implement various CMLs. While similar approaches like the Meta Object Facility (MOF) [6] and ECore [39] exists, we develop our approach based on the ADONIS meta meta model [29]. Therefore, the modeling ontology is designed based on the meta constructs *model type*, *class*, *relation class* and *attribute* which are stored as OWL classes. Model types modeling languages as instances like BPMN or UML which are the base for classes and relation classes. Classes have instances like events, activities or gateways depending on the modeling language. Relation classes have the connectors as instances. Classes and relation classes can have attributes related through OWL object properties like *hasShape*, defining the shape, e.g. rectangle or circle, of the modeling construct. The attributes for the graphical representation are used for creating the toolbar of the plugin. For relation classes the OWL object properties *hasStartClass* and *hasEndClass* can be defined and with object property assertions related to class elements. Based on this generic template the properties of different CMLs can be defined and are created while initializing the toolbar view.

Several options for the technical implementation were considered. Three general implementation options seemed reasonable: a. extend an ontology editor, b. extend a modeling editor or c. create a proprietary software combining the ontology and the modeling concepts. As implementing a new proprietary software is an incredible complex task demanding deep knowledge on either ontology development and modeling, the problem limited to the decision on extending either an ontology editor or a modeling editor. Assuming in our conceptual idea the use of two ontologies, one for the domain knowledge and one for the modeling knowledge, it has been chosen to follow option a. by extending an ontology editor.

The decision for a plugin based on Protégé platform was made, because of its wide use and its openness for external plugins. The plugin is implemented with JAVA, the Graphical User Interfaces (GUI) are designed and realized with JavaFX[4]. They were designed sparse intentionally, but in a way, they could be extended easily. The plugin is embedded in Protégé as separate Protégé Tab containing a *toolbar view* and a *canvas view*(figure 1). The toolbar view contains the modeling language classes and relations as specified in the modeling ontology, which can be added to the canvas view in terms of the modeling process.

The two ontologies approach demands a way of mapping the domain ontology and the modeling ontology for connecting the domain knowledge and the modeling representation. Hinkelmann et al. [25] distinguishes between automated and human-interpreted semantically enrichment of meta models. Our approach uses the human-interpreted approach for adding semantical information in the modeling process. While creating a new modeling entity a name entered and a class of the domain ontology must be selected. In the process of adding the graphical

[4] The plugin is available at `https://github.com/benediktreitemeyer/onbacomo`

element to the canvas, additionally an OWL Individual is created, added to the domain ontology and classified regarding the class that was selected during the modeling procedure. As well, it is annotated with the suiting graphical representation. When a relation between two model elements is created, a name must be entered and an object property of the domain ontology has to be selected. While being added to the canvas an object property assertion to the start element of the relation is made which enables reasoning on the model elements.

In this section we discussed the key assumptions the plugin is based on and described the key aspects its implementation. Focus has been on the ontological foundation and the modeling mechanism which is used for a manual mapping between the domain ontology and the modeling ontology. In section 5 an initial use case is introduced, explaining the application of the plugin in real world context.

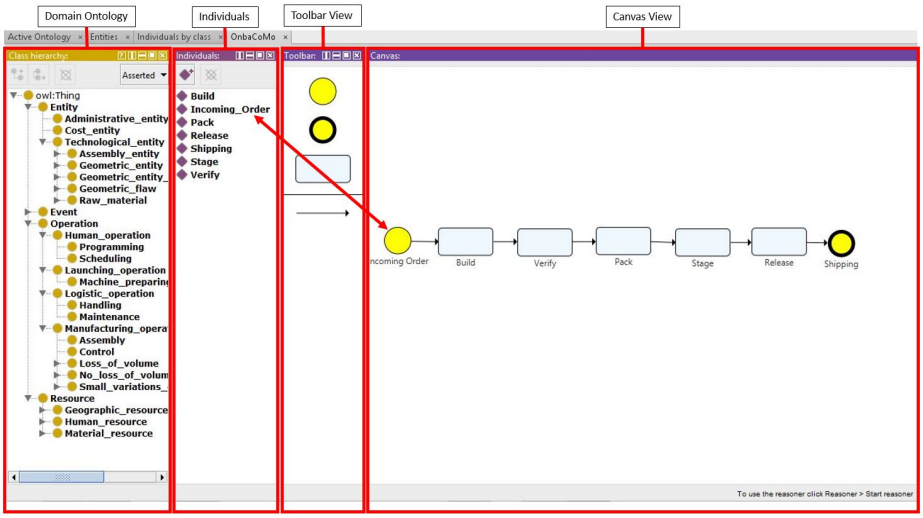

Fig. 1. Layout and GUI of the plugin.

5 Use Case

In this section the application of the plugin is shown with the help of a use case. Therefore, a modeling ontology, which is constructed with *BPMN*, and a domain ontology *Manufacturing's Semantics Ontology (MASON)* [32] are used to show the modeling of a business process. For the practical use case BPMN is used as modeling language and implemented based on the generic modeling ontology template. BPMN is chosen for this case, because it has emerged as a standard notation in process modeling, e.g. for work flows as manufacturing processes, and

high-level system design. Beneath its practical use, the comprehensive amount of scientific research on BPMN, e.g. on its formal semantics [49], is an additional reason for its use in the use case. Even though various ontologies representing BPMN exist, the template is used because of its ability to be used with different modeling languages.

In terms of the categorization of ODCM seen in section 2, the implementation could be categorized in the dynamic perspective. Currently the classes start event, end event and task are implemented with their attributes. They can be connected with the relation class subsequent. Using these elements simple BPMN models can be created. Domain ontology in the use case is the MASON ontology. It is an OWL ontology created with the target to gain a common semantic net for manufacturing domain. Its practical use is for example automatic cost estimation [32]. An assembly process was modeled, based on the Value-Chain Group's VRM (Value Reference Model) framework.The tasks build, verify, package, stage, and release are modeled as BPMN tasks and connected with the subsequent relation class (see the model in Figure 1). In addition, a start event and an end event are added. Starting with these initial conditions,

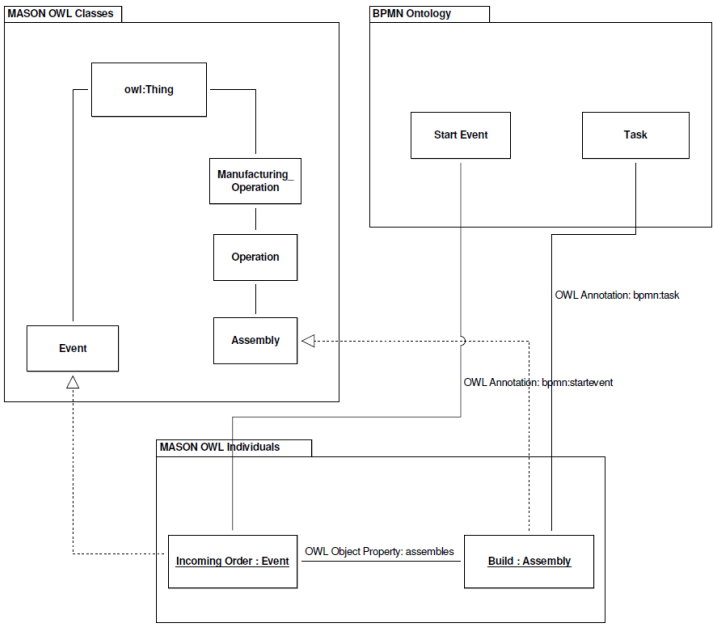

Fig. 2. MASON OWL Classes and Individuals mapped to BPMN Ontology Classes.

the process is modeled. While modeling each of the BPMN tasks and events are mapped manually to MASON classes and the BPMN subsequents are mapped to MASON object properties. While modeling, OWL individuals are created based

on the BPMN elements OWL entities and mapped through annotation properties to the selected MASON OWL entities. This enables the reasoning. Figure 2 shows the conceptual background of this solution for the BPMN start event *Incoming Order* and the BPMN task*Build*. *Incoming Order* is an instance of the MASON class *Event*, while *Build* is an instance of the MASON class *Assembly*. The instances are connected through the OWL object property *assembles*. Furthermore, both instances have OWL annotations referring to the used BPMN element. Based on this construct reasoning can be performed leading to the resulting Protégé views in figure 3.

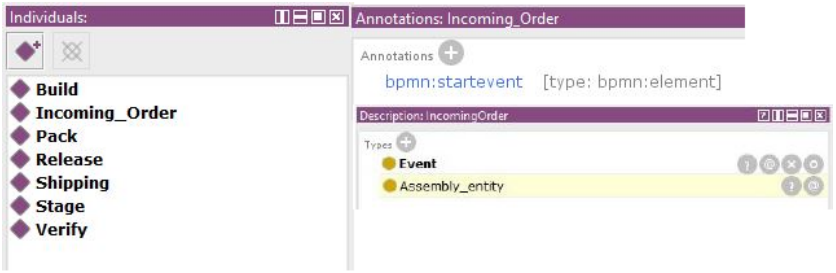

Fig. 3. Created Individuals, Annotations and Reasoning in the Use Case

With the implementation and the use case we showed that an approach based on domain and modeling ontologies is feasible. In the next section we will additionally discuss the benefits and drawbacks of the solution.

6 Discussion

In this section the *benefits* and *drawbacks* are discussed. One of the key benefits is that the knowledge gap on domain knowledge and modeling knowledge are closed through commonly used ontologies. Furthermore, the domain and modeling knowledge is not only human-processable but becomes machine-processable. Especially in terms of the machine-interpretation of models this is important and leads to a third important benefit, the ability to apply reasoning on the models and gain new knowledge. We expect finding additional benefits while extending the approach in terms of reasoning.

As the work is still in progress, there are some drawbacks. Currently, the implementation has only been tested with BPMN as modeling language. Implementing other modeling languages can verify the generic approach of the work. Additionally, the current implementations lack BPMN elements like gateways. In regard to the two different reasoning types of Wang et al. [46], we only proved the ability of ontology reasoning, but not the ability of user-defined reasoning.

7 Conclusion and Further Research

In this paper we introduced an ODCM approach based on an ontological foundation of the domain knowledge and the modeling knowledge. For this purpose, we initially differentiated the concepts of visualization and modeling in general and enterprise contexts. The technical feasibility of the approach has been shown by implementing it as a Protégé plugin and applying it to a use case.

The focus of our future research is on completing the BPMN implementation. Further research will include, implementing and testing the plugin with another modeling language for evaluating the generality of the approach. Implementing and evaluating of reasoning e.g. syntax-checking should be investigated. Conceptually, evaluating the quality of the developed ontologies should be performed. Lastly, a more complex case study which is performed with different editors is necessary to evaluate the advantages of the plugin in regard to other modeling editors.

References

1. Antoniou, G., Van Harmelen, F.: Web ontology language: Owl. In: Handbook on ontologies. Springer (2004)
2. Azzini, A., Braghin, C., Damiani, E., Zavatarelli, F.: Using semantic lifting for improving process mining: a data loss prevention system case study. In: SIMPDA. pp. 62–73 (2013)
3. Benevides, A.B., Guizzardi, G.: A model-based tool for conceptual modeling and domain ontology engineering in ontouml. In: International Conference on Enterprise Information Systems. pp. 528–538. Springer (2009)
4. Bork, D.: A Development Method for the Conceptual Design of Multi-View Modeling Tools with an Emphasis on Consistency Requirements. PhD Thesis, University of Bamberg (2015)
5. Bork, D., Fill, H.G.: Formal aspects of enterprise modeling methods: a comparison framework. In: System Sciences (HICSS), 2014 47th Hawaii International Conference on System Science. pp. 3400–3409. IEEE (2014)
6. Brockmans, S., Volz, R., Eberhart, A., Löffler, P.: Visual modeling of owl dl ontologies using uml. In: International Semantic Web Conference. pp. 198–213. Springer (2004)
7. Bubenko, J., Persson, A., Stirn, J.: User guide of the knowledge management approach using enterprise knowledge patterns, deliverable d3, ist programme project hypermedia and pattern based knowledge management for smart organisations, project no. ist-2000- 28401. Royal Institute of Technology (2001)
8. Burton-Jones, A., Storey, V.C., Sugumaran, V., Ahluwalia, P.: A semiotic metrics suite for assessing the quality of ontologies. Data & Knowledge Engineering **55**(1), 84–102 (2005)
9. Card, S.K., Shneiderman, B., MacKinlay, J.D.: Readings in information visualization: using vision to think. Morgan Kaufmann (1999)
10. COMPOSER, T.: Topbraid composer 2007 features and getting started guide version 1.0, created by topquadrant. US. US (2007)
11. Dietz, J.: What is Enterprise Ontology? Springer (2006)
12. Fill, H.G.: Visualisation for semantic information systems. Gabler (2009)

13. Fill, H.G.: SeMFIS: A Tool for Managing Semantic Conceptual Models (2012)
14. Fill, H.G., Karagiannis, D.: On the Conceptualisation of Modelling Methods Using the ADOxx Meta Modelling Platform. Enterprise Modelling and Information Systems Architectures **8**(1), 4–25 (2013)
15. Fill, H.G.: On the conceptualization of a modeling language for semantic model annotations. In: CAiSE Workshops. pp. 134–148. Springer (2011)
16. Fox, M.S., Barbuceanu, M., Gruninger, M.: An organisation ontology for enterprise modelling: preliminary concepts for linking structure and behaviour. In: 4th Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises. pp. 71–81. IEEE (1995)
17. Fox, M.S., Gruninger, M.: Ontologies for enterprise modelling. In: K., K., J.G., N. (eds.) Enterprise Engineering and Integration. pp. 190–200. Springer (1997)
18. Frank, U.: Visual languages for enterprise modelling (1999)
19. Gruber, T.R.: Toward principles for the design of ontologies used for knowledge sharing? International journal of human-computer studies **43**(5-6), 907–928 (1995)
20. Guarino, N.: Formal ontology and information systems. In: Guarino, N. (ed.) Proceedings of FOIS'98. pp. 81–97. IOS Press (1998)
21. Guizzardi, G.: Ontological foundations for conceptual modeling with applications. In: Ralyté, J., Franch, X., Brinkkemper, S., Wrycza, S. (eds.) Advanced Information Systems Engineering. pp. 695–696. Springer (2012)
22. Guizzardi, G., Wagner, G.: Using the unified foundational ontology (ufo) as a foundation for general conceptual modeling languages. In: Theory and Applications of Ontology: Computer Applications. pp. 175–196. Springer (2010)
23. Guizzardi, G., Wagner, G., Almeida, J.P.A., Guizzardi, R.S.: Towards ontological foundations for conceptual modeling: the unified foundational ontology (ufo) story. Applied ontology **10**(3-4), 259–271 (2015)
24. Harel, D., Rumpe, B.: Meaningful modeling: What's the semantics of "semantics"? IEEE Computer **October 2004**, 64–72 (2004)
25. Hinkelmann, K., Albayrak, M., Kritikos, K., Kurjakovic, S., Lammel, B., Woitsch, R.: Modelling framework for bpaas. In: CloudSocket (2015)
26. Hinkelmann, K., Laurenzi, E., Martin, A., Thönssen, B.: Ontology-based metamodeling. In: Business Information Systems and Technology 4.0, pp. 177–194. Springer (2018)
27. Horrocks, I., Patel-Schneider, P., Boley, H., Tabet, S., Grosof, B., Dean, M.: Swrl: A semantic web rule language combining owl and ruleml. W3C Member Submission **21**, 79 (2004)
28. Iacob, M., Jonkers, H., Lankhorst, M., Proper, H.: ArchiMate 1.0 Specification. Van Haren Publishing (2009)
29. Junginger, S., Kühn, H., Strobl, R., Karagiannis, D.: Ein geschäftsprozessmanagement-werkzeug der nächsten generation - adonis: Konzeption und anwendung. Wirtschaftsinformatik **42**(5) (2000)
30. Karagiannis, D., Mayr, H.C., Mylopoulos, J.: Domain-specific conceptual modeling - Concepts, Methods and Tools. Springer (2016)
31. Kifer, M., Lausen, G.: F-logic: a higher-order language for reasoning about objects, inheritance, and scheme. ACM SIGMOD Record **18**(2), 134–146 (1989)
32. Lemaignan, S., Siadat, A., Dantan, J.Y., Semenenko, A.: Mason: A proposal for an ontology of manufacturing domain. In: Workshop on Distributed Intelligent Systems. pp. 195–200. IEEE (2006)
33. Maier, R.: Knowledge Management Systems: Information and Communication Technologies for Knowledge Management. Springer (2004)

34. McBride, B.: The resource description framework (rdf) and its vocabulary description language rdfs. In: The handbook on ontolgies in information systems. Springer (2003)
35. Musen, M.A.: The protégé project: a look back and a look forward. AI matters **1**(4), 4–12 (2015)
36. Mylopoulos, J.: Conceptual modeling and telos. In: Conceptual Modelling, Databases and CASE: An Integrated View of Information Systems Development. pp. 49–68. Wiley (1992)
37. Opdahl, A.L., Berio, G., Harzallah, M., Matulevičius, R.: An ontology for enterprise and information systems modelling. Applied Ontology **7**(1), 49–92 (2012)
38. Sandkuhl, K., Fill, H.G., Hoppenbrouwers, S., Krogstie, J., Matthes, F., Opdahl, A., Schwabe, G., Uludag, Ö., Winter, R.: From expert discipline to common practice: A vision and research agenda for extending the reach of enterprise modeling. Business & Information Systems Engineering **60**(1), 69–80
39. Staab, S., Walter, T., Gröner, G., Parreiras, F.S.: Model driven engineering with ontology technologies. In: Reasoning Web International Summer School. pp. 62–98. Springer (2010)
40. Strahringer, S.: Ein sprachbasierter Metamodellbegriff und seine Verallgemeinerung durch das Konzept des Metaisierungsprinzips. Modellierung **98**, 15–20 (1998)
41. Studer, R., Benjamins, R., Fensel, D.: Knowledge engineering: Principles and methods. Data & Knowledge Engineering **25**, 161–197 (1998)
42. Uschold, M.: Building ontologies: Towards a uni ed methodology. In: Proceedings of 16th Annual Conference of the British Computer Society Specialists Group on Expert Systems. Citeseer (1996)
43. Uschold, M., King, M., Moralee, S., Zorgios, Y.: The enterprise ontology. The knowledge engineering review **13**(1), 31–89 (1998)
44. Verdonck, M., Gailly, F.: Insights on the use and application of ontology and conceptual modeling languages in ontology-driven conceptual modeling. In: ER Conference. pp. 83–97. Springer (2016)
45. Wand, Y., Weber, R.: An ontological model of an information system. IEEE transactions on software engineering **16**(11), 1282–1292 (1990)
46. Wang, X., Zhang, D.Q., Gu, T., Pung, H.K.: Ontology based context modeling and reasoning using owl. In: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops. IEEE (2004)
47. Weiten, M.: Ontostudio as a ontology engineering environment. In: Davies, J.F., Grobelnik, M., Mladenic, D. (eds.) Semantic Knowledge Management. pp. 51–60. Springer (2009)
48. Wilde, T., Hess, T.: Forschungsmethoden der wirtschaftsinformatik. Wirtschaftsinformatik **49**(4), 280–287 (2007)
49. Ye, J., Sun, S., Song, W., Wen, L.: Formal semantics of bpmn process models using yawl. In: 2008 Second International Symposium on Intelligent Information Technology Application. vol. 2, pp. 70–74. IEEE (2008)