Enterprise-Specific Ontology-Driven Process Modelling

Nadejda Alkhaldi[1(✉)], Sven Casteleyn[1,2], and Frederik Gailly[3]

[1] Vrije Universiteit Brussel, Pleinlaan 2, 1050 Brussels, Belgium
nadejda.alkhaldi@vub.ac.be, sven.casteleyn@uji.es
[2] Universitat Jaume I, Av. De Vicent Sos Baynat s/n, 12071 Castellon, Spain
[3] Ghent University, Tweekerkenstaart 2, 9000 Ghent, Belgium
frederik.gailly@ugent.be

Abstract. Different process models are created within an enterprise by different modelers who use different enterprise terms. This hinders model interoperability and integration. A possible solution is formalizing the vocabulary used within the enterprise in an ontology and put this ontology as bases for constructing process models. Given that an enterprise is an evolving entity, the ontology needs to evolve to properly reflect the domain of the enterprise. This paper proposes an enterprise-specific ontology-driven process modelling method which tackles the two aforementioned issues by assisting the modeller in creating process models using terminology from the ontology and simultaneously supporting ontology enrichment with feedback from those models. When the modeller creates a model, matching mechanisms incorporated in the method are working together to suggest a list of ontological concepts that have a high potential to be useful for a particular modelling element. When the model is created, its quality is first evaluated from different perspectives to make sure that it can be used within the enterprise, and second to discover whether its feedback can be useful for the ontology. When the feedback is extracted, the proposed method incorporates guidelines on how to use this feedback.

Keywords: Business process modeling · Enterprise ontology · Ontology-driven modelling · BPMN · UFO

1 Introduction

When different models within an enterprise are created by different modelers, integrating those models is hard. A possible solution for this integration problem is providing modelers with a shared vocabulary formalized in an ontology [1] and [2]. Over the last 30 years, different domain ontologies have been developed which describe the concepts, relations between concepts and axioms of a specific domain. In a business context, a particular type of domain ontologies are so-called enterprise ontologies. They describe the enterprise domain and consequently provide enterprise domain concepts that can be reused by different enterprises. Example of enterprise ontologies include the Enterprise Ontology [3], TOVE [4] and the Resource Event Agent enterprise ontology [5]. Two different approaches have been proposed to incorporate enterprise ontologies into the modeling process. Some authors consider enterprise ontologies to be reference models

J. Cordeiro et al. (Eds.): ICEIS 2014, LNBIP 227, pp. 472–488, 2015.
DOI: 10.1007/978-3-319-22348-3_26

that support the creation of different kind of models. For instance, [6] suggests developing the Generic Enterprise Model as an ontology that is later used as a reference for creating both data and process models. Other authors developed an enterprise-specific modelling language which is based on the concepts, relations and axioms described in the enterprise ontology [7].

In this paper we focus on using enterprise-specific ontologies (ESO) during the development of business process models. Enterprise-specific ontologies are domain ontologies that differ from enterprise ontologies in the fact that their Universe of Discourse is a specific enterprise, rather than the enterprise domain. They may have their origin in an established domain ontology or in an enterprise ontology, but their main goal is describing the concepts, relations and axioms that are shared within a particular enterprise. Enterprise-specific ontologies are getting increasingly important in the context of data governance and knowledge representation [1]. Supporting tools, such as IBM InfoSphere[1] or Collibra Enterprise Glossary[2] allow enterprises to specify their own enterprise glossary/ontology. Such an enterprise-specific ontology, once available, can subsequently be deployed to help enterprise modellers in creating compatible, enterprise-specific models, such as requirements, data or process models. This paper focuses on business process models. Additionally the ESO ontology needs to be maintained and enriched while the enterprise evolves. Enriching ESO from the process models is very practical because it will reflect processes that were introduced recently within the enterprise, or processes that where adjusted.

The work described in this paper is a process model-specific instantiation of a framework for ontology-driven enterprise modeling aimed to facilitate model construction based on an enterprise-specific ontology on one hand, and support enterprise-specific ontology creation and evolution based on feedback from the modelling process on the other hand. This framework also proposes criteria to evaluate the quality of resulting models to ensure that their feedback is potentially useful. To illustrate our work, we use the Unified Foundational Ontology as core ontology, OWL as ontology representation language and BPMN as business process modelling language.

2 Enterprise-Specific Ontology Engineering and Process Modelling Method

The method that is proposed in this paper in an instantiation of the meta-method which can be used for different enterprise modeling languages and which was proposed in previous work [8]. As displayed in Fig. 1, the ontology and modeling method consists of two parallel cycles, which in turn consist of different phases. For a general description of the different phases we refer to [8]. In this paper we will focus on describing the different phases specifically for process modeling using BPMN.

[1] http://www-01.ibm.com/software/data/infosphere/.
[2] http://www.collibra.com/.

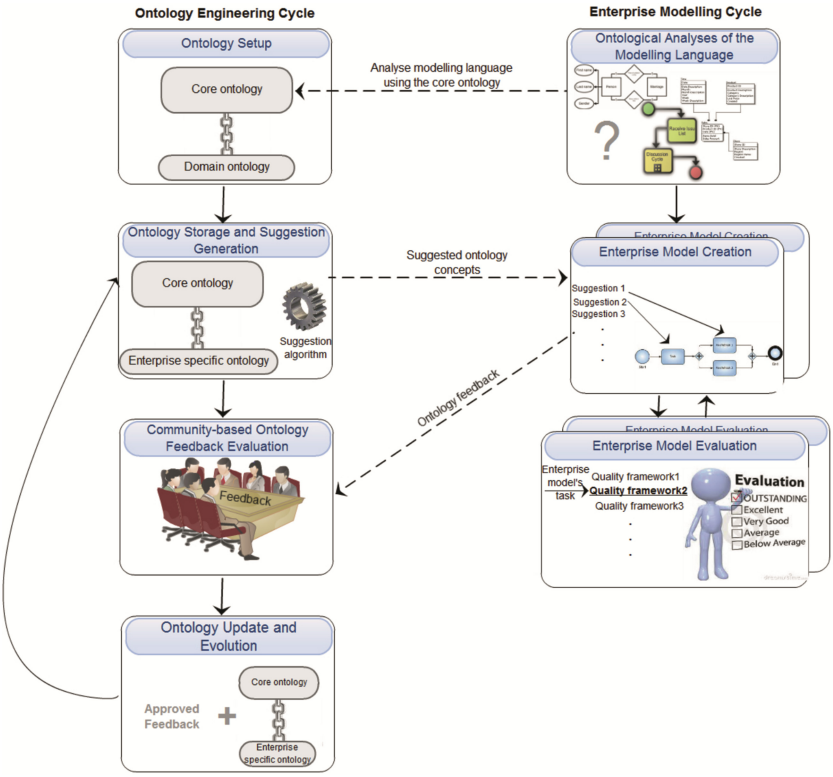

Fig. 1. Enterprise specific ontology and modeling method.

2.1 Ontology Setup Phase

This is the first phase of the Ontology Engineering cycle. It includes pre-processing of the selected enterprise-specific ontology (ESO) so that matching mechanisms can be applied in the later phases to derive useful suggestions. Pre-processing in this case implies mapping concepts from the selected ESO to a core ontology. A core ontology is an ontology that describes universally agreed upon, high level concepts and relations, such as objects, events, agents, etc. [9]. Constructs of the selected modelling language will also be mapped to this core ontology in the next phase of the proposed method. The core ontology thus forms a bridge between the ESO and the selected modeling language, and incorporates shared semantics.

The Unified Foundational Ontology (UFO) was selected as a core ontology in this work for three reasons: 1/ the benefits of grounding domain ontologies in UFO are well motivated [10], and several such UFO-grounded domain ontologies are available, e.g., [11] 2/ UFO is specifically developed for the ontological analysis of modelling languages, and 3/ BPMN was chosen as modelling language in this work and analysis of BPMN using UFO is available in literature [12].

UFO has different layers of which here only UFO-U is used, as this is sufficient for finding construct-based matches between the selected modeling language (explained later) and ESO. However, it can be further investigated if using (full) UFO is beneficial to refine the proposed algorithms. The top level element in UFO-U is a Universal. It represents a classifier that classifies at any moment of time a set of real world individuals and can be of four kinds: Event type, Quality universal, Relator universal and Object type.

For the purpose of this demonstration, we have selected as ESO an existing ontology in financial domain.[3] This ontology contains static concepts related to finance, such as Branch, Customer, Loan, Insurance, etc., which can be used as a reference for models constructed in the financial domain. A sample of the mapping between the ESO and UFO-U is presented in Table 1.

Table 1. Mappings between ESO and UFO-U.

ESO concept	UFO-U	ESO concept	UFO-U
AddedValue	Quality_ Universal	Liability	Relator Universal Mediates Customer and mediates Branch
Adminstrative	Role_Type	Loan	Relator Universal Mediates Customer and mediates Branch
Asset	Mixin Type	Mortgage-Loan	Relator Universal Mediates Customer and mediates Branch
Branch	Base_Type	Castomer	Mixin Type

2.2 Ontological Analyses of the Modeling Language

The selected modeling language also needs to be analyzed because the core ontology forms a intermediary through which modeling constructs are mapped to the concepts of ESO. Although our meta-method support any kind of core ontology, it is important that the used modelling languages are analysed using the selected core ontology because the suggestion generation process relies hereupon. Table 2 represents mappings between BPMN and UFO-U.

[3] http://dip.semanticweb.org/documents/D10.2eBankingCaseStudyDesignandSpecificationofApplicationfinal.pdf.

Table 2. Mappings between BPMN and UFO-U.

BPMN construct	UFO-U	BPMN construct	UFO-U
Pool	Object type	End event	Event Type
Lane	Object type	Event noun	Base type, Mixin type, Relator universal
Task Noun	Relator universals or quality universal	Condition Exclusive Gateway	Quality universal
Noun Sub Process	Relator universals or quality universal	Data object	Relator universal,, Base type
Start event	Event Type	Message flow label	Relator universal
Intermediate event	Event Type		

2.3 Ontology Storage and Suggestion Generation Phase

Every time the modeler places a modeling element on the canvas, several matching mechanisms are cooperating in order to rank the ESO concepts to display the most relevant of them on the top of a suggestions list. Given the potentially extensive amount of ESO concepts, relevance ranking of suggestions is a critical feature.

Depending on the type of modelling construct that is added, the position of the construct relevant to other elements (i.e., its neighborhood) in the model, and the label entered by the modeller, the order of the suggestion list is prioritized so that ontology concepts with a higher likelihood to be relevant in the current context appear first. To achieve this, four different suggestion generation mechanisms are used. These mechanisms are partly inspired by ontology matching techniques [13], but are specifically focused to fit within our framework, where the semantics of the modelling language can be exploited.

Every matching technique calculates a relevance score (between 0 and 1) for each ESO concept, which is stored. Subsequently, the overall relevance score is calculated using a weighted average of all individual scores. This corresponds to the formula below:

$$ConceptRelevanceScore = \frac{S_s W_s + S_{syn} W_{syn} + S_c W_c + S_{nb} W_{nb}}{W_s + W_{syn} + W_c + W_{nb}}$$

Where:
$S_s W_s$: the score and weight of string match
$S_{syn} W_{syn}$: the score and weight of synonym match
$S_c W_c$: the score and weight of construct match
$S_{nb} W_{nb}$: the score and weight of neighborhood based match

The weights for each matching mechanism are thus configurable. In our demonstration (see Sect. 3), we assigned a higher weight to string matching as we expect that, within a particular enterprise context, a (quasi) exact string match has a high possibility of representing the intended (semantic) concept. The lowest weight is assigned to construct matching, because it typically matches very broadly, and thus delivers a large amount of suggestions. Further experimentation with the distribution of weights over the individual relevance scores should be performed to determine an optimal overall score calculation. This is considered future work. As a final result, the suggestion list, a descending ordered list of ESO concepts according to relevance, is generated and presented to the modeller. In the next four subsections the different mechanisms are described in more detail. The implementation of the mechanisms can be consulted via our Github repository: https://github.com/fgailly/CMEplusBPMN.

2.3.1 String Matching Mechanism

The goal of the string matching algorithm is to find ESO concepts whose label is syntactically similar to the label of modeling elements entered by the modeller. If these two strings are syntactically the same, there is a high possibility that they have the same semantics, especially as both reside within the same enterprise and business context.

Currently Jaro-Winkler distance [14] is used to calculate the edit distance between the given BPMN element label, and the label of each concept in the enterprise-specific ontology. The Jaro-Winkler distance was chosen because this hybrid technique takes into account that the text entered by the modeller can contain spelling errors, and additionally favours matches between strings with longer common prefixes (i.e., a substring test, which is very useful in our context because matching is executed each time a character is added to the label). Jaro-Winkler distances are between 0 (no similarity) and 1 (exact match), and are thus immediately useable as a relevance score.

2.3.2 Synonym Matching Mechanism

The synonym matching mechanism aims to detect synonyms of the given BPMN element label (or part of it) in the ESO. To realize this, WordNet [15] is used. WordNet is an online lexical database that organizes English nouns, verbs, adjectives and adverbs into sets of synonyms (so-called synsets). It is thus ideal to find synonyms. For each synonym of the modeling element label, the previously described string matching algorithm is performed on all ESO concepts, thereby generating a relevance score between 0 (no match) and 1 (exact synonym match).

2.3.3 Construct Matching Mechanism

This matching mechanism operates based on the mapping performed in "ontology set up" and "ontological analyses of the modeling language" phases described previously. Consequently, BPMN modeling constructs can be mapped to ESO concepts through the UFO core ontology. During the matching process, this mechanism assigns a score of 1 to all ESO concepts mapped to the same UFO-U construct as the modeling language construct created. All the other ESO concepts are assigned relevance score of 0. In our implementation, the mappings between the ESO and UFO,

and BPMN and UFO, are each represented in an OWL file. OWL reasoning is then used to perform the mappings between the ESO concepts and BPMN constructs based on the two aforementioned OWL files.

2.3.4 Neighborhood – Based Matching Mechanism

Neighborhood-based mechanism calculates relevance scores for ESO concepts based on the location of the newly added modeling element, the type of modeling element that is added, and the relationships between the ESO concepts corresponding to the modeling elements surrounding the newly added element. The neighborhood of a BPMN element is determined by the connectivity objects (i.e. sequence flow, message flow, association), and which pool (or lane) the BPMN element is located in. In other words, for every element we can determine which pool or lane it is a part of, and which other element(s) is/are connected to this element using either a sequence, message flow or association. Next, the relationships (which are specified in terms of the UFO-U relationships through the ESO-UFO-U mappings) between the ESO concepts are exploited. According to [10] there are two types of relations: formal and material. A *formal relation* between entities holds directly, without any further intervening individuals. A *material relation* has material structure by itself. It includes relations such as *working at, being treated at*, etc. Entities related by this type of relation are mediated by individuals called relators. In Sect. 3 we will demonstrate how the UFO-U relators can be used to suggest concepts from the ESO.

Finally, using both the relative position of the new element and the material relations between the ESO concepts, the element neighborhood-based mechanism can now derive relevance scores for ESO concepts in relation to some BPMN modeling elements (for examples, see Sect. 3):

1. To create a pool construct when another pool already exists, the suggestions (relevance score 1) are UFO-U object types that are related by a material relationship with the ESO concept with which the existing pool(s) was/were annotated (i.e., the ontology annotations of the pool(s)).
2. To create a lane construct within a pool, the suggestions (relevance score 1) are UFO-U role types that are related by a material relationship with the ontology annotation of the pool(s)
3. To create a message construct that results in transmitting a message between a task or event of a pool and another pool, the suggestions (relevance score 1) are UFO-U relators mediating material relations connecting objects that annotate respectively the noun of the task and the ontology annotation of the pool.
4. To create a conditional gateway, there are two ways to derive suggestions (both receive relevance score 1):

- ESO concepts annotated by the task label preceding the gateway. This can work very well for tasks that are performing evaluation or calculation, after which the gateway is used to make a decision based on the results. In this case, the condition on the gateway will use the same concept as used in the task. This concept is most likely to be a quality, especially if the task at hand is performing calculations. Nevertheless, it can also be a relator, such as for example verifying if the contract is ok or not.

- Qualities associated with the UFO-U object type annotation of the pool where the gateway is located. Or UFO-U qualities associated with UFO-U object types participating in material relations with Object type annotation of the pool where the gateway is located.

5. For creation of a task construct, the suggested concepts are most likely to be related through material relations to the pool where the task is located. The suggestions can be either quality types of the concept annotating that pool or relators mediating those material relations.

2.4 Enterprise Model Creation Phase

During this phase the modeler proceeds with creating the process model utilizing suggestions derived by the matching mechanisms described in the previous section. With every modeling element placed on the canvas the modeler is advised to select ESO concept from the suggestions list to annotate the modeling element. This annotation implies maintaining a link between the modeling element and the corresponding concept of the ESO. With the help of the annotation, the modeling element is semantically connected to the ESO concept even if they have different labels. In our implementation, the annotation is realized by creating an OWL file where a URI of the OWL ESO concept is added to the corresponding modeling element using the OWL annotation mechanism. A portion of the annotation OWL file is presented below. It shows that pool construct "Participant_1" has a label "Customer" and is annotated by "Customer" concept of the bank ontology (our ESO).

```
<ClassAssertion>
<Class IRI = "http://www.mis.ugent.be/ontologies/bpmn.owl#Pool"/>
<NamedIndividual IRI = "#Participant_1"/>
</ClassAssertion>
<AnnotationAssertion>
<AnnotationProperty abbreviatedIRI = "rdfs:isDefinedBy"/>
<IRI > #Participant_1 </IRI>
<Literal datatypeIRI = "http://www.w3.org/2001/XMLSchema#string">
http://www.mis.ugent.be/ontologies/bank#Customer
</Literal>
</AnnotationAssertion>
```

2.5 Enterprise Model Evaluation Phase

Within the Enterprise-Specific Ontology and Modeling framework, the model quality evaluation phase has two main goals: (1) it ensures that the model can be used within the enterprise and (2) that the feedback extracted from the creation of the model is useful and potentially worth incorporating into the ESO. Literature provides a plethora of frameworks for quality evaluation of different kinds of models. For an overview of how process model quality can be evaluated we refer to [16]. A well-known scheme for

classifying quality dimensions is the Lindland et al. framework [17] which makes a distinction between syntactic, semantic and pragmatic quality dimensions. *Syntactic quality* implies correspondence between the model and the modelling language. *Semantic quality* measures how compliant the model is to the domain. And *pragmatic quality* is the correspondence between the model and the user interpretation of it.

The first goal of the model evaluation phase can be satisfied by focusing on measures that fall within the syntactic and pragmatic dimension because for the model to be used within the enterprise it needs to be correct and understandable. The second goal can be achieved by focusing on measures that fall within the semantic quality dimension. More specific the developed process model has to reflect the process it was designed for. Hence, the three quality dimensions postulated by [17] are well suited for this phase of the meta-method.

2.5.1 Syntactic Quality

This dimension stipulates that the created model must be syntactically correct in order to be used within an enterprise. If the model has syntactic flows, its process cannot be correctly implemented. Syntactic quality is achieved using syntactic correctness criterion. According to [18] verification of the syntactic quality of a process models focuses on two properties: static property and behavioral property. Static property is the related to the elements of the model and how they are used and connected. For example, in BPMN it is not allowed to have sequence flow between two pools. The static syntactic quality can be verified by the modeling tool itself while the model is being created. Behavioral property relates to the behavior of the process model. For example, the process cannot reach a deadlock and proper process completion is guaranteed. This is evaluated automatically by computer programs after the model is created.

2.5.2 Semantic Quality

Semantic quality of a process model is typically evaluated by means of completeness and validity measures. Completeness is defined in [16] as a degree to which a model has all the necessary and relevant information. Following the work of [19] completeness of a process model can be measured by: 1. Counting the number of items in BPMN model that do not correspond to the description of the actual process 2. counting the number of requirements that are present in model description, but are not reflected in the model itself. It is advisable to document the process to be constructed so that this document can serve as a reference for model evaluation. But when no description is available, the quality evaluation metrics can be executed by another stakeholder familiar with the process, or by the modeler himself. According to [18] the model is valid when all its statements are correct and relevant to the problem. In order to verify validity, one must know the meaning of modeling elements and the process that the model is representing.

2.5.3 Pragmatic Quality

Pragmatic quality is about the correspondence between the process model and users' comprehension of it. This quality dimension is not relevant when the modeler himself is the only user of the model because he obviously will understand the model he made.

But if the model will be presented to other stakeholders, it is very important that they understand it.

The literature contains several propositions on how to evaluate understandability. [16] proposes using structural complexity metrics suggested by [20] (which is specific to BPMN) as an indication of the degree of model understandability. If the model is complex, it is likely to be less understandable. In the context of the proposed method it is possible to use a complexity metric to make sure that the model is not exceeding complexity limits.

2.6 Community-Based Ontology Feedback Evaluation Phase

The method only proceeds to this phase if the model satisfies the expected semantic quality, meaning that it correctly reflects the process it was designed for, and all the statements in the model are valid within the process. When the process model is complete, an OWL file representing this model and the ESO concepts selected for the model annotation is stored. This file is processes in order to extract any possible feedback which is potentially useful and can be incorporated into the ESO. A possible feedback is a listing of the elements from the model that were not annotated by ESO concepts. As they were not annotated, the reason might be that there is no equivalent for them in the ESO. This feedback is made available for other community members and is subject to discussion, until finally a consensus is reached whether or not the proposed change(s) should be included (such as new concepts added) in the new version of the ontology. The community will discuss the proposed concepts such as their usage, definition and usefulness within the enterprise. Community members are other people working in the same business domain. As community members are typically not co-located at the same physical location, and we are aiming to progressively reach a consensus about what is needed by the community, the Delphi approach [21] is used. This approach is perfectly suited to capture collective knowledge and experience of experts in a given field, independently of their location, and to reach a final conclusion by consensus. More specifically, consensus is reached by commenting on the feedback in 3 cycles. Three cycles were chosen because studies show that most changes in responses occur in the first 2 rounds [21]. In every cycle comments are assigned a score, and when all three cycles are accomplished, a decision is taken whether to incorporate feedback or discard it. Only in case the community is not able to reach a consensus, the final decision is made by community members with a high level of trust. A system for assigning trust credits to community members is foreseen.

If negotiation upon the feedback progresses slowly, the process may be terminated without accomplishing predefined number of cycles. In this case highly trusted, authorized community members are responsible to make a decision.

2.7 Ontology Update and Evolution Phase

After the feedback verification is performed, ontology expert incorporate its results into the enterprise ontology. It is worth mentioning that ontology experts do not interfere in feedback verification. Their mission is limited solely to incorporating the final results

into the ontology in a syntactically correct way. Once a considerable amount of feedback is incorporated, a new ontology version is proposed. The new ontology incorporates new concept/relationships, updates lacking/incomplete ones, and/or removes irrelevant once, as the domain evolves or new insights are reached by the expanding community.

3 Demonstration

This section illustrates the suggestions generation phase of the method explained in the previous sections by means of a lab demonstration in which the modeller constructs a process model in BPMN notation in the financial domain using an existing financial domain ontology[4] as enterprise-specific ontology.

Once the ESO is grounded in the core ontology (as mentioned, this only needs to be done once, and can subsequently be re-used for any model created within the enterprise), the modeller can start creating the process model. He selects a construct to be added, places it on the canvas and starts typing the construct's desired name. As he selects the construct, and as he is typing, the mechanisms described in the previous section derive suggestions from the ESO and present them to the modeller. If an ESO concept in the suggestion list appropriately corresponds to the intention of the modeller for this particular BPMN construct, he selects this concept, and the BPMN construct is (automatically) annotated with the chosen ESO concept.

The process model to be created in our lab demonstration represents the loan application assessment process in a bank, and is taken from [22]. By using an existing specification, we avoid bias towards our method. The process starts when the loan officer receives a loan application from one of the bank's customers. This loan application is approved if it passes two checks: the first check is the applicant's loan risk assessment, which is done automatically by the system after a credit history check of the customer is performed by a financial officer. The second check is a property appraisal check performed by the property appraiser. After both checks are completed, the loan officer assesses the customer's eligibility. If the customer is found to be not eligible, the application is rejected. Otherwise, the loan officer starts preparing the acceptance pack. He also checks whether the applicant requested a home insurance quote. If he did, both the acceptance pack and the home insurance quote are sent to the applicant. If the insurance was not requested, only the acceptance pack is sent. The process finally continues with the verification of the repayment agreement.

Figure 2 represents the BPMN model of the loan application process. Constructs that are surrounded by a thick red square are annotated with ESO concepts.

[4] http://dip.semanticweb.org/documents/D10.2eBankingCaseStudyDesignandSpecificationofApplicationfinal.pdf.

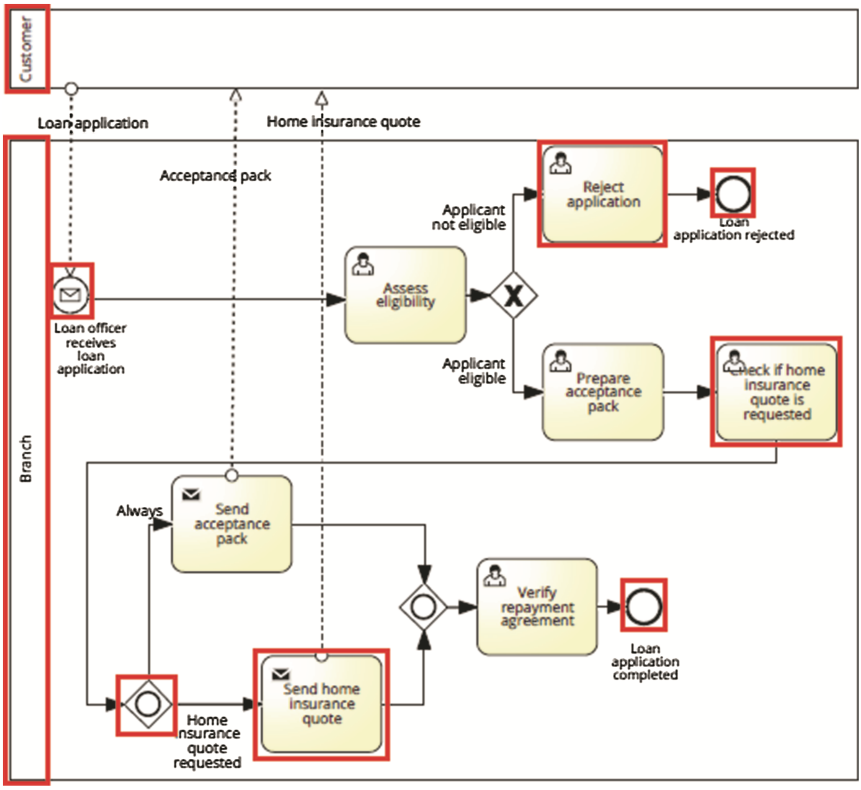

Fig. 2. BPMN model describing loan application.

Adding Branch Pool: The modeller selects the pool construct to be created. Based on the construct matching mechanism all ESO concepts corresponding to UFO-U object type are given a relevance score of 1 for this matching mechanism. Among those concepts the modeller can find Branch which is classified as UFO-U base type. If there is already a "Customer" pool, based on the construct neighbourhood matching mechanism, this case corresponds with rule 1. As the already existing pool is the Customer pool, the mechanism looks for ESO concepts related to the Customer concept through material relationships. There is only one concept satisfying this requirement: Branch. As a result, the Branch concept is listed in the beginning of the suggestion list, as it scored for both the construct and neighbourhood matching mechanisms (and no other concept scored equal or higher). Note that in this scenario, string and synonym matching cannot contribute to the overall relevance score yet, as the modeller did not (yet) type any label.

Adding Message Flow "Loan Application": According to the construct matching mechanism, message flow corresponds to the relator universal. Therefore, all ESO concepts corresponding to the UFO-U relator universal will be selected. Those concepts are: Channel, loan, mortgage loan, current mortgage loan, future mortgage loan, invoice,

liability, payment. For the element neighbourhood- based matching technique this situation resolves under rule 2.

In our enterprise-specific ontology, all relator universals are mediating the same two concepts Branch and Customer. Therefore, the results delivered by this suggestion generation technique are the same as the results delivered by construct matching. In this case, the previously mentioned suggestions all have equal overall score, and can thus not be prioritized. We therefore present them alphabetically. The modeller may select a concept from the list (i.e., "loan"), or, in case the list is too long, start typing any desired label (e.g., "loan" or "credit"). This triggers the string- and synonym-based matching mechanisms, both of which prioritize the concept Loan, which consequently appears on top of the suggestion list, and is selected by the modeller to annotate the loan application message flow.

Adding Reject Application Task: The modeller selects the BPMN task construct, and subsequently the construct matching mechanism assigns a high relevance score to all ESO concepts that correspond to UFO-U quality and relator universals as suggestions for the task noun. A list of relator universals is mentioned in the previous example; a list of quality universals is very exhaustive and is thus not mentioned here. The second mechanism, element neighbourhood based matching, applies rule 3: the task at hand is located in the Branch pool, so this matching mechanism suggests all the ESO concepts corresponding to UFO-U relator universals related to Branch concept in the ESO, and UFO-U object types mediated by those relators (all with relevance score 1). The Loan concept is a relator universal, and therefore received relevance score of both matching mechanisms; it therefore appears on the top of the suggestions list.

Adding "Home Insurance Quote is Requested" Gateway: In the last scenario, the modeller draws an inclusive decision gateway on the canvas. Based on construct matching mechanism, all quality universals will be assigned a priority score of 1. The element neighbourhood-based mechanism classifies this situation under rule 4, which suggests ESO concepts that were used to annotate a task construct preceding the gateway. In this case it is the "check if home insurance quote is requested" task, which is annotated with the Home Insurance concept. This concept thus receives relevance score 1 for the gateway, and is prioritized in the suggestion list. It perfectly matches our needs.

To start the discussion, we note that the scenarios elaborated here were chosen to illustrate the more complicated cases. As a result, string- and synonym-based matching mechanism are underrepresented. Evidently, when no or few BPMN elements are already on the canvas, neighbourhood-based matching will be unable to sufficiently differentiate between potential suggestions (as in scenario 2), and string- and synonym-matching will become important. Equally, when the modeller has a certain label already in mind, string and synonym matching will dominate the suggestion list, as the modeller is typing the label he had in mind. Having made this comment, we note that in general, it was possible to derive suggestions based on the construction and element neighbourhood matching mechanisms for all model constructs for which the related concepts existed in the ontology. In fact, as can be seen in the scenarios, construct and

neighbourhood based matching complement each other well. The majority of the concepts required by the model, but missing from the ontology were also correctly classified under the assumptions of neighbourhood-based matching mechanism, and would have been assigned a high relevance score if they would have been present in the ontology. However, there was one case where the neighbourhood-based matching mechanism was not very accurate. While creating the last message flow "Home insurance quote", based on the second assumption of the neighbourhood-based matching mechanism and the construct matching mechanism, relator universals must be suggested. But in reality, it was annotated with a quality universal HomeInsurance, instead of a relator. Further fine-tuning of the suggestion generation mechanism should avoid this type of mismatches.

The lab demonstration was used here to demonstrate viability of our method, to detail the different steps and provide a concrete case. It shows that the suggestion generation algorithm indeed provides useful suggestions to the modeller, and allows (automatic) annotations of the model, thereby semantically grounding them and facilitating model integration. It needs to be mentioned that the lab demonstration was done using a single modeller, and that therefore the aforementioned positive indications of using our method cannot be statistically proven. We are currently performing a more elaborate empirical validation, where a group of test users is divided in three different groups: one group is given an ESO and our method, the second group is only given the ESO but without support of our method, and the last group is not given an ESO and thus needs to model without any ontology or method support. The experiment is specifically designed to show the impact of our model on modelling efficiency, consistency in the use of terminology, and the semantic grounding of the resulting models.

After the model is created, it is time to perform model quality evaluation. Syntactic quality is measured by counting the amount of violations of BPMN syntax. From the static perspective, the model in Fig. 2 is syntactically correct which is expected as the modeling tool itself prevents some basic violations. From the behavioral aspect it is important to look into different scenarios of model completion. This model will always reach a valid completion independently of which paths are selected at the gateway. There are no tasks in the model that can never be executed. To evaluate semantic quality, we need to look back to the model description in the beginning of this demonstration section. Because within the context of this method, this description represents the domain to which the model needs to correspond. There are two requirements in the description that are not reflected in the process model in Fig. 2. The first requirement is: "applicant's loan risk assessment, which is done automatically by the system", and the second requirement is: "credit history check of the customer is performed by a financial officer". Those requirements need to be incorporated before the feedback from the model is taken.

To evaluate the pragmatic quality, structural complexity is calculated based on measures presented in [16] presented in Table 3. According to [23] those measures are directly related to understandability. The thresholds suggested for those values are: number of nodes between 30 and 32, Gateway mismatch is between 0 and 2, depth is 1, connectivity coefficient is 0.4. With those values the model is considered to be 70% understandable. The values obtained in evaluating the model in Fig. 2 are close to the proposed threshold, and therefore the model is potentially understandable.

Table 3. Structural complexity measurements of the model in Fig. 2.

Metric	Result
Number of nodes in the model	15
Gateway mismatch (sum of gateway pairs that do not match with each other)	1
Depth (maximum nesting of structure blocks)	0
Connectivity coefficient (Ratio of the total number of arcs in a process model to its total number of nodes)	13/15 = 0.9

4 Conclusions and Future Work

This paper presented an overview of an enterprise-specific ontology-driven process modeling method. On the one hand, this method improves semantic consistency of process models by relying in ESO in model construction. On the other hand, it supports the evolvement of the ESO according to practical needs of the enterprise by taking feedback from the process models. While constructing models, the modeler is aided by a list of suggestions extracted from the ESO. ESO concepts are ranked in a suggestions list using four matching mechanisms. Two of them, the string and synonym matching mechanisms, are based on the label of the newly created BPMN element, which is systematically compared with concepts in the ESO. The other two, namely construct matching and neighbourhood-based matching, depend on the type of the BPMN construct and the position (relative to other modelling elements) where it is added.

Another benefit of the proposed method is that it facilitates maintenance and improvement of the ESO by means of feedback from the process models. This feedback is only accepted if the resulted model properly reflects the process it is representing. The proposed method incorporates guidelines on model quality evaluation. When a model is created, an OWL file containing all modeling elements is stored. This file is processed by ontology expert and all the elements representing potential feedback. This feedback is subject to community discussion and if approved, it will be incorporated in the new ontology version.

Future work will follow different directions. Concerning model creation based on the ESO, first, we are currently performing further empirical validation of the benefits of our method. Second, suggestions towards the ontology (e.g., missing concepts) based on the modelling process, and subsequent community-based ontology evolution, needs to be explored. Finally, we also plan to apply the method for other modelling languages (i.e. i*, KAOS), and using other core ontologies. Concerning ESO maintenance and amelioration, we are planning to set up a forum where the community will discuss model feedback. Clear definition of guidelines for discussion and voting is also very essential.

References

1. Bera, P., Burton-Jones, A., Wand, Y.: Guidelines for designing visual ontologies to support knowledge identification. MIS Q. **35**(4), 883–908 (2011)
2. Di Francescomarino, C., Tonella, P.: Supporting ontology-based semantic annotation of business processes with automated suggestions. In: Halpin, T., Krogstie, J., Nurcan, S., Proper, E., Schmidt, R., Soffer, P., Ukor, R. (eds.) Enterprise, Business-Process and Information Systems Modeling. LNBIP, vol. 29, pp. 211–223. Springer, Heidelberg (2009)
3. Uschold, M., King, M., Moralee, S., Zorgios, Y.: The enterprise ontology. Knowl. Eng. Rev. Spec. Issue Putting Ontol. Use **13**(1), 31–89 (1998)
4. Fox, M.S.: The TOVE project: towards a common-sense model of the enterprise. In: Belli, F., Radermacher, F.J. (eds.) Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, pp. 25–34. Springer, Germany (1992)
5. Geerts, G.L., McCarthy, W.E.: An accounting object infrastructure for knowledge based enterprise models. IEEE Intell. Syst. Their Appl. **14**, 89–94 (1999)
6. Fox, M., Gruninger, M.: On ontologies and enterprise modelling. In: Kosanke, K., Nell, J.G. (eds.) Enterprise Engineering and Integration, pp. 190–200. Springer, Heidelberg (1997)
7. Sonnenberg, C., Huemer, C., Hofreiter, B., Mayrhofer, D., Braccini, A.: The REA-DSL: A domain specific modeling language for business models. In: Mouratidis, H., Rolland, C. (eds.) CAiSE 2011. LNCS, vol. 6741, pp. 252–266. Springer, Heidelberg (2011)
8. Gailly, F., Casteleyn, S., Alkhaldi, N.: On the symbiosis between enterprise modelling and ontology engineering. In: Ng, W., Storey, V.C., Trujillo, J.C. (eds.) ER 2013. LNCS, vol. 8217, pp. 487–494. Springer, Heidelberg (2013)
9. Doerr, M., Hunter, J., Lagoze, C.: Towards a core ontology for information integration. J. Digit. Inf. **4**(1) (2006)
10. Guizzardi, G., Wagner, G.: What's in a relationship: an ontological analysis. In: Li, Q., Spaccapietra, S., Yu, E., Olivé, A. (eds.) ER 2008. LNCS, vol. 5231, pp. 83–97. Springer, Heidelberg (2008)
11. Barcellos, M.P., Falbo, R., Dal Moro, R.: A well-founded software measurement ontology. In: 6th International Conference on Formal Ontology in Information Systems (FOIS 2010), vol. 209, pp. 213–226 (2010)
12. Guizzardi, G., Wagner, G.: Can BPMN be used for making simulation models? In: Barjis, J., Eldabi, T., Gupta, A. (eds.) EOMAS 2011. LNBIP, vol. 88, pp. 100–115. Springer, Heidelberg (2011)
13. Euzenat, J., Shvaiko, P.: Ontology Matching. Springer, Berlin (2013)
14. Winkler, W.: String comparator metrics and enhanced decision rules in the Fellegi-Sunter model of record linkage. In: Proceedings of the Section on Survey Research Methods, American Statistical Association (1990)
15. Miller, G.: WordNet: a lexical database for English. Commun. ACM **38**(11), 39–41 (1995)
16. Sánchez-González, L., García, F., Ruiz, F., Piattini, M.: Toward a quality framework for business process models. Int. J. Coop. Inf. Syst. **22**(01), 1350003 (2013)
17. Lindland, O., Sindre, G., Sølvberg, A.: Understanding quality in conceptual modeling. IEEE Softw. **11**(2), 42–49 (1994)
18. Reijers, H.A., Mendling, J., Recker, J.: Business Process Quality Management. In: vom Brocke, J., Rosemann, M. (eds.) Handbook on Business Process Management 1, pp. 167–185. Springer, Berlin (2010)
19. Moody, D.L.: Metrics for evaluating the quality of entity relationship models. In: Ling, T.-W., Ram, S., Li Lee, M. (eds.) ER 1998. LNCS, vol. 1507, pp. 211–225. Springer, Heidelberg (1998)

20. Rolón, E., Ruiz, F., Garcia, F., Piattini, M.: Applying software metrics to evaluate business process models. CLEI-Electron. J. **9**(1) (2006)
21. Gupta, U.G., Clarke, R.E.: Theory and applications of the delphi technique: a bibliography (1975–1994). Technol. Forecast. Soc. Chang. **53**, 185–211 (1996)
22. Dumas, M., La Rosa, M., Mendling, J., Reijers, H.: Fundamentals of Business Process Management. Springer, Heidelberg (2013)
23. Sánchez-González, L., Garcia, F., Mendling, J., Ruiz, F.: Quality assessment of business process models based on thresholds. In: Meersman, R., Dillon, T.S., Herrero, P. (eds.) OTM 2010. LNCS, vol. 6426, pp. 78–95. Springer, Heidelberg (2010)