Automatic Generation of Ontologies from Business Process Models

Lukas Riehl Figueiredo and Hilda Carvalho de Oliveira
Institute of Geosciences and Exact Sciences, São Paulo State University (Unesp), Rio Claro, Brazil
lukasrie@ibilce.unesp.br, hildaz@rc.unesp.br

Keywords: Business Process Model, BPMN, XPDL, Ontology, OWL, SPARQL.

Abstract: Business process models are used in organizational environments for a better understanding of the interactions between the different sectors and the interdependencies between processes. However, business process models may present legibility problems and navigation difficulties as they become extensive. The representation of implicit knowledge is complex, as well as the interdependencies are not always easy to be understood. The use of ontologies has opened a complementary perspective to provide processes with machine-accessible semantics. Ontologies contribute to the conceptualization and organization of the embedded and unstructured information that is present in the business processes and that must be explored. The ontologies are used to structure the implicit knowledge that is present in the business processes, allowing the understanding by machine. They also facilitate the sharing and reusing of knowledge by various agents, human or artificial. In this context, this work presents a systematic process to generate an ontology from a business process model in BPMN, allowing to query information about the model. For this, the PM2ONTO tool was developed, aiming to generate the ontology in OWL automatically and to provide predefined queries, elaborated with SPARQL.

1 INTRODUCTION

Business process models are used by business professionals for a better understanding of the interactions between the different sectors that make up an organizational environment and the interdependencies between processes. Despite the importance of business processes, these processes are not always part of the organization's knowledge base, being available only in business process modeling tools (Guido et al., 2016).

On the other hand, the understanding of the business domain is also essential for professionals in the area of Information Technology (IT). They are responsible for software solutions and for automating the organization's process flows. Software requirements must be adherent to the business environment, both in relation to processes performed in the environment and to the vocabulary and language of the business domain. However, many times the gap between the vision/language of business professionals and the ones of IT is often too great. Przybylek (2014) already observed that the Requirements Engineering process is hampered by the lack of communication due to the use of many different notations between the business area and IT.

The BPMN notation (Business Process Model Notation) defines many graphical features for visual representation of the elements of a business process model. In addition, BPMN is a standard of the OMG (Object Management Group) consortium and has been increasingly used worldwide. On the other hand, Correia and Abreu (2015) emphasize that the BPMN specification is a relatively complex document for process modelers.
Another important aspect is that the BPMN specification does not provide a standard for documenting business process models.

Business process models may present legibility problems and navigational difficulties as they become extensive. The representation of the implicit knowledge in its elements is complex, and interdependencies are not always easy to understand. Often, the reading of these models is limited to the understanding of the graphic elements that compose the models and their relations.

Some approaches of Business Process Management follow an ontological perspective, aiming to provide processes with semantics accessible by machine (GÓMEZ-PÉREZ, 2010). According to Gábor and Kõ (2016), the ontologies contribute to the conceptualization and organization

Figueiredo, L. and Oliveira, H.
Automatic Generation of Ontologies from Business Process Models.
In *Proceedings of the 20th International Conference on Enterprise Information Systems (ICEIS 2018) - Volume 2*, pages 81-91
ISBN: 978-989-758-298-1

of the embedded and unstructured information that is present in the business processes and that must be explored. Ontologies are used to structure the implicit knowledge that is present in the business processes, allowing the understanding of this knowledge by machine. They also facilitate the sharing and reuse of knowledge by various agents, whether human or artificial.

In this direction, this paper aims to propose a systematic process for the automatic generation of an ontology from a business process model in BPMN notation v2.0. The ontology can map explicit and implicit knowledge in the model, evidencing existed relations and interdependencies, as well as facilitating queries by software systems.

The systematic process requires that the business process model be exported to the language XPDL (XML Process Definition Language) v2.2. This language is specified by the organization WfMC (Workflow Management Coalition) and provides a XML-structured file containing the process definitions. It is important to emphasize that the business process models that will be used to generate the ontologies must follow some good modeling practices and present documentation of their elements.

The expected objective with the application of the systematic process is that the BPMN graphic elements and the documentation associated to them are properly mapped to classes of the generated ontology, evidencing the relations and interdependencies among them.

In addition, this paper presents the PM2ONTO tool (Process Model To Ontology), to generate the ontology and allow predefined queries about the ontology. The XPDL files which correspond to the business process model are used as inputs for the generation of the ontology in the language OWL (Web Ontology Language), using the framework Apache JENA (Apache Software Foundation, 2017). The queries about ontology are coded with the language SPARQL. The Protégé system is used for the analysis and visualization of the generated ontologies.

To better understand the context of this work, Section 2 presents some works related to the generation of ontologies from business process models. The intention is to show other directions and the markable differences with the proposal of this work. Section 3 presents the steps of the systematic process for automatic ontology generation. In Section 4 an overview of PM2ONTO tool is presented, as well as the process for mapping the XPDL elements to the ontology classes. This section also presents SPARQL language queries defined for the PM2ONTO tool. Section 5 presents the application of the systematic process and the PM2ONTO tool on a real business process model. The final considerations and the future works are presented in Section 6.

2 RELATED WORK

Some works in the literature also present approaches for the creation of ontologies from business process models in BPMN. Among them: Haller et al. (2007), Missikoff et al. (2010), Fanesi et al. (2015), Ternai et al. (2016) and Guido et al. (2016). Only Haller et al. (2007) and Missikoff et al. (2010) also export the models to the XPDL language, but they use versions prior to XPDL 2.2. In addition, the BPMN elements are not mapped individually to the ontology, as in this work. On the other hand, Guido et al. (2016) define only basic relations between the elements of BPMN and interdependencies are not deeply explored. However, in this work will be defined all the possible relations to be extracted from the context where each element is in the model.

Haller et al. (2007) apply ontology concepts to the XPDL language (version 2.0), in order to add semantic value in the process definitions. The authors present a tool that performs the automatic conversion of process instances to the ontology, called oXPDL. The BPMN elements are used to define ontology's properties, such as functional, control, informational, organizational, and operational aspects.

In another direction, Missikoff et al. (2010) propose the creation of a framework based on the BPAL (Business Process Abstract Language) language, in order to add semantics to business process schemas through formalisms and rules. This language is used in conjunction with domain ontologies using OWL (Web Ontology Language) to capture the knowledge of business processes, resulting in an enterprise knowledge base. It is observed, the construction of the knowledge base does not use logical formalisms but a direct mapping from the BPMN elements to an ontology in OWL.

Fanesi et al. (2015) aim to add greater semantic value to business process models using the definition of an unified ontology (composed of several layers) based on the OWL-FA language (an extension of OWL-DL). The first layer of the ontology is composed by metamodels, which contain the ontology classes. The second layer is formed by instances of concepts that compose the ontology in the first layer. Instances of this second layer make up the third one. The intention of the authors is to allow

ontology queries on several aspects of a business process model. The ontology defined by the authors should classify each class relative to a BPMN element, as well as its instance, in a layer. This approach classifies concepts as classes, instances, or even then as classes and instances at the same time. However, in this work, all the elements of a model are defined as classes, favoring their categorization in the ontology, as well as the definition of relations and queries.

Guido et al. (2016) define a metamodel for BPMN notation v2.0 composed of classes and properties in OWL. This metamodel covers the same BPMN elements as this work, with the following major classes: graphical_element (subclasses swimlane, flow_object, connecting_object, artifact and data), lane, pool and supporting_element (subclasses event_detail and participant). Disjunction constraint classes are defined from object instances and properties. However, the relations defined for BPMN elements are limited and are not self-explanatory, which makes it difficult to understand the interdependencies existing in business process models. In this paper, this type of problem has been mitigated, since it is not limited only to the direct relations existing in the business process model; relations encompass all elements belonging to interdependencies.

Ternai et al. (2016) use the business process model converted to an XML file using the BOC ADONIS modeling platform (Boc Group, 2013). From this XML file, the authors generate a process ontology of a metamodel with the following classes: Actor, IT_System, Data_Object, Process_Step e Decision_Point. The conversion of a business process model into an ontology is accomplished through a script in the XSLT language (Extensible Stylesheet Language for Transformation). However, this script must follow a predefined structure; otherwise not all the elements of the model will be mapped to the ontology. Instead of using scripts, this work uses a metamodel that can be applied to any XPDL file to obtain the classes that will compose the ontology because it conforms to the existing definitions in the XPDL files.

3 SYSTEMATIC PROCESS FOR AUTOMATIC GENERATION OF THE ONTOLOGY

This section presents the systematization of the steps for the automatic generation of an ontology in OWL from a business process model in BPMN v2.0. The objective is that the graphic elements of BPMN and the documentation associated to these elements be properly mapped to ontology classes, evidencing the relations and interdependencies between these elements.

In this context, Section 3.1 presents some criteria in order to execute the mapping to the ontology suceessfully. Section 3.2 presents the process to automatically generate the ontology that represents a business process model.

3.1 Criteria to Business Process Models

The elements of BPMN considered in the business process models for this work are: activities, sub-process, events, gateways (exclusives, inclusives and parallels), artifacts (groups and annotations), data objects, data stores, extended attributes, pools, lanes, sequence flows and message flows.

The graphic elements of the model must be associated with textual documentation, according to the format of the selected BPMN modeling system (there is no standardized way to structure the organization of textual information of the BPMN elements).

In addition, the models must satisfy the following criteria, based on good modeling practices:

- Every process must have an unique event, start event and at least one end, with names "Start" and "End" respectively;
- A process must contain at least one participant;
- The textual documentation of each BPMN element must contain at least one of its descriptive properties: name, description or documentation;
- The name of an activity must use a verb in the infinitive;
- The type of activities must always be defined;
- An activity must be performed by at least one actor;
- Activity actions must not be assigned as gateway names (gateways represent the routing logic after a decision making);
- Sequence flows of an exclusive gateway must be named, except in the case that there are only two flows representing "Yes" or "No". In this case, only one of the names can be explicit and the other can be deduced;

– Sequence flows of an inclusive gateway must be named;
– Activities that are preceded by a parallel gateway must be connected at the end by another parallel type gateway;
– Activities that are preceded by an inclusive gateway must be connected at the end by another gateway of the inclusive type;

In this work, extended attributes add important value to the ontology. These attributes are not available in all BPMN modeling systems, but are very useful and their use is encouraged. Extended attributes assist in documenting the business process model and in the identification of some software elements for this work. The Fig. 1 shows the extended attribute to the activity "Validate Invoice". (These attributes will be categorized in the ontology as functional requirements ("FR"), nonfunctional requirements (NFR) or business rules ("BR"), according to the Requirements Heuristics defined by Nogueira and Oliveira (2017).

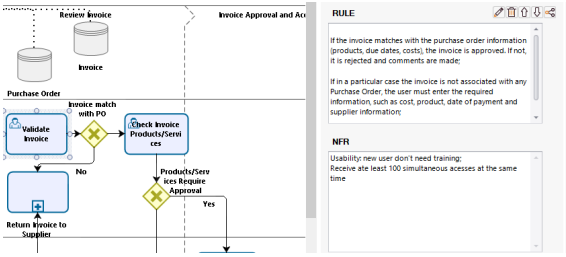

Figure 1: Example of extended attribute defined for an activity.

3.2 Ontology's Generation

The process for the generation of the ontology counts on three steps, that must be executed sequentially in this order:

– **Step 1**: the business process model in BPMN notation v2.0 must be exported to the XPDL v2.2 language through a system for business process modeling. Considering that the model has n sub-processes, this export will generate n + 1 XPDL files, one for each sub-process and one for the main process;
– **Step 2**: the PM2ONTO ("Process Model to Ontology") tool should be used to generate the ontotology automatically, using the XPDL files generated in the previous step as input. The result will be an ontology defined in the OWL language that represents the original business process model. The PM2ONTO tool executes the following steps in this order, as illustrated in Fig. 2:

2.1. Reading and mapping XPDL elements for classes defined from a metamodel that contemplates the elements of BPMN considered in this work;
2.2. Generation of the concepts of the ontology and of the relation properties of them from the classes generated in step 2.1;
2.3. Storage of the ontology content generated in a database and provision of the ontology OWL file for download.

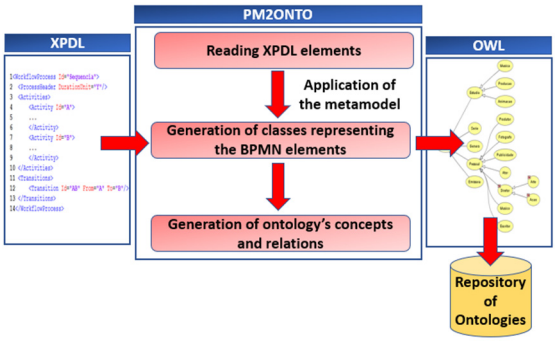

Figure 2: Overview of the PM2ONTO tool.

4 PM2ONTO TOOL

The PM2ONTO tool was developed in Java language v1.8 to be used in the Web browser. In addition to the generation of ontologies, PM2ONTO allows queries about the generated ontologies (Fig. 3). Queries are defined in the SPARQL language and can extract basic information from the elements as well as their relations within a business process model.

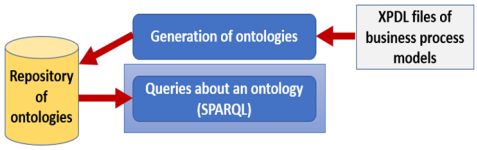

Figure 3: Functionalities of the PM2ONTO tool.

The Apache Jena freeware framework was used to generate ontologies and queries. The Hibernate framework (an implementation of Java Persistence API) was used for persistence and object-relational mapping of the database; this allowed the storage of the generated ontologies in a database. The database system used was MySQL v. 5.7. Protégé system (v.

5.0.0 or higher) can be used for the purpose of visualizing and validating the generated ontologies.

Section 4.1 presents how is the process for mapping elements of the XPDL files to the classes of the generated ontology. On the other hand, Section 4.2 presents the predefined queries in the current version of the PM2ONTO tool.

4.1 Mapping Metamodel's Classes to Ontology's Classes

A metamodel was defined to enable the mapping of the BPMN elements to the concepts of the ontology. Fig. 4 shows part of the metamodel in UML (Unified Modeling Language) notation, while Table 1 shows the relations used in the metamodel. In Fig. 4 it can be observed that all classes, with the exception of the Actor class, have at least the following basic attributes: identifier, name, description, documentation, and type. The Actor class does not contain the type attribute, but contains the other basic attributes mentioned.

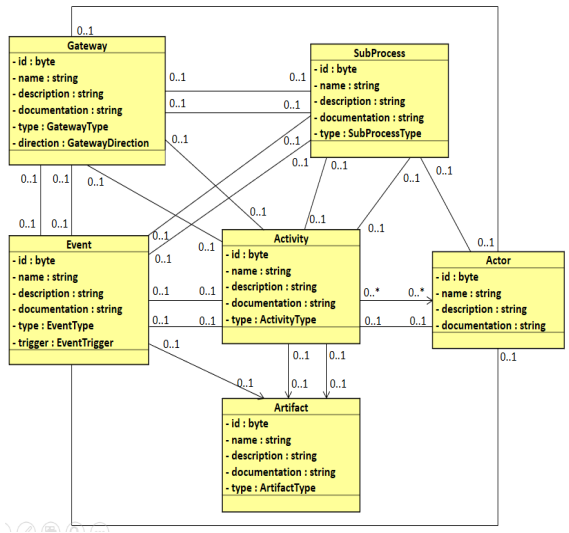

Figure 4: Part of the metamodel's classes.

The metamodel is formed by 19 classes divided into three categories:

- **Main classes**, which correspond to the following elements of BPMN: Process, Activity, SubProcess, Event, Gateway, Actor and Artifact;
- **Auxiliary or enumerator classes**, that define the types and other specificities of BPMN elements: SubProcessType, ActivityType, GatewayType, EventType, ArtifactType, GatewayDirection and EventTrigger;
- **Relation classes**, which define the relation between two or more BPMN elements: SuceededBy, UsesInput, ProducesOutput, ExchangesMessageWith, and ExecutedBy.

Table 1: Relations in the metamodel.

Relation	Source element	Target Element
Is suceeded by	SubProcess, Activity, Event, Gateway	SubProcess, Activity, Event, Gateway
Is executed by	Activity	Actor
Exchanges message with	SubProcess, Activity, Event, Gateway, Actor	SubProcess, Activity, Event, Gateway, Actor
Uses input	Activity	Artifact
Produces output	Activity, Event	Artifact

It is important to mention that the information defined in the XPDL file about the relative positioning (coordinates) of the elements in the business process model were not considered in this work.

When XPDL files related to a business process model are entered as input into the PM2ONTO system, the elements of each XPDL file are analyzed for the metamodel application and generation of the ontology. For this, Table 2 shows how the correspondences between the XPDL elements and the classes of the metamodel were made. The other XPDL elements were not considered for class mapping in this work.

The elements identified as "Performer" were mapped to the "Actor" class of the metamodel. The "DataStore", "DataObject" and "Artifact" elements have been mapped to the Artifact class of the Metamodel . The "Activity" elements of each process flow can define one of the following classes of the metamodel: Activity, Event or Gateway.

The actors were associated to the activities they perform through the mapping of the "Performer" elements when found inside the "Activity" element; in this case, the relations between the actors and their activities were mapped for the class of the metamodel ExecutedBy.

The relations between data objects and/or data stores with activities or events, defined by the "DataAssociation" and "DataStoreReference" associations, have been mapped to the UsesInput and ProducesOutput classes.

When the Activity, Event and Gateway classes present extended attribute stores, they must be mapped to the ExtendedAttribute class. Extended attributes with name "RN", "RF" or "RNF" were defined as business rule, functional requirement or

Table 2: Relations among the XPDL elements and classes of the metamodel.

XPDL element	Metamodel´s class
WorkflowProcess	Process
Performer	Actor
Activity (with element Task defined)	Activity
Activity (with element Event defined)	Event
Activity (with element Route defined)	Gateway
ActivitySet	SubProcess
DataStore	Artifact
DataObject	Artifact
Artifact	Artifact
Artifact (with ArtifactType defined as Annotation)	Annotation
Artifact (with ArtifactType defined as Group)	Group
ExtendedAttribute	ExtendedAttribute
Transition	SuceededBy
MessageFlow	ExchangesMessageWith
Activity (with element Performer defined)	ExecutedBy
Data Association	UsesInput/ProducesOutput
Data Store Reference	UsesInput/ProducesOutput

nonfunctional requirement, respectively. In the sequence, the "ActivitySet" elements were analyzed and mapped as sub-processes; for each occurrence of an "ActivitySet", the previous steps of mapping BPMN elements to the classes were repeated, to define all the elements belonging to the sub-processes.

The transitions (element "Transition") contained in the XPDL files were mapped to the SuceededBy class of the metamodel, which is formed by two attributes: source element and destination element. Message flows ("MessageFlow" element) have been mapped to the ExchangesMessageWith class, which also contains the "source element" and "destination element" attributes.

The classes of the metamodel define the concepts and the relations that compose the ontology. The ontology classes are categorized into levels according to the BPMN elements to which they refer (Fig. 5). Each class defined from the metamodel is analyzed and then included as a new class in the ontology, being classified according to the defined categorization.

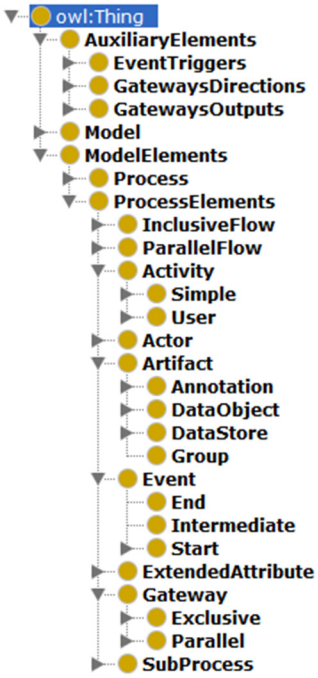

Figure 5: Categorization of classes for the ontology.

The first level classes are:

- **AuxiliaryElements:** composed of the following subclasses: EventTriggers, GatewayDirections and GatewaysOutputs. The class EventTriggers defines the possible triggers for the event triggering. The GatewayDirections class defines the possible directions of a Gateway (divergent or convergent). The GatewaysResponses class defines the possible gateway return types;
- **Model**: is the class that defines the main information of the original business process model;
- **ModelElements**: this class includes the classes of the business processes itselves (Process subclasses) and their elements (ProcessElements subclasses); this class of the ontology is composed by the classes of all the elements of a model. This class is related to the Model class by means of the isComposedBy object property, indicating that a business process model is characterized by a set of elements.

The ProcessElements class is present on the second level and contains the subclasses: Activity, Actor, Artifact, Event, ExtendedAttribute, Gateway, and SubProcess. It is important to mention that the Activity, Event, Gateway and SubProcess classes are categorized according to the type they belong to. The Gateway class, for example, is categorized as

Exclusive, Parallel, and Inclusive because only the exclusive, parallel, and inclusive gateways are considered in this work.

The following data properties were defined for the ontology with the objective to describe better the classes: id, name, description and documentation, as well as the boolean properties isBusinessRule, isFunctionalRequirement and isNonFunctionalRequirement. When these boolean properties have the value "true", it means that the class that contains this property is identified as a business rule, functional requirement or non-functional requirement.

The ontology classes are related through object properties. In this work, these properties relate two or more classes of the ontology and are divided into two groups:

- **Basics**: define the direct relations that BPMN elements can assume (Table 3). They are formed from the transitions, message flows, and associations present in XPDL files. Inverse properties have been defined for some of the basic object properties, such as isPerformedBy, whose inverse property is isExecutorOfActivity;
- **Advanceds**: aim to provide a better understanding of the interdependencies of the sub-processes, activities, gateways and events present in a business process model. Table 4 shows the four advanced properties defined and which are explained below.

The advanced object property isActivatedWhenEventIsTriggered is defined for the BPMN elements that are preceded by events. The intention is to indicate that these elements are invoked from the trigger caused by the related event.
The isExecutedWhen [Activity name] OutputIs object property defines the relation between two BPMN elements interconnected by a routing of an exclusive gateway. This relation is intended to make the connection between the element preceding the exclusive gateway and the element after that gateway clearer. The element that succeeds the exclusive gateway is executed after the routing of the execution result of the first element. It is important to note that the "[Activity name]" part of the property identifier refers to the name of the activity that is executed before the routing performed by the exclusive gateway. Thus, the constraint for the application of this property is the existence of an activity succeeded by an exclusive gateway, provided that the outputs resulting from the routing of that gateway are properly named.

Table 3: Basic object properties defined for the ontology.

Property	Domain	Range
isComposedBy	Model	ModelElements
isPartOfProcess	SubProcess, Activity, Event, Gateway, Actor, Artifact	Process
isPartOfSubProcess	Activity, Event, Gateway, Actor, Artifact	SubProcess
isPartOfGroup	SubProcess, Activity, Event, Gateway, Actor, Artifact	Group
isSuceededBy	SubProcess, Activity, Event, Gateway	SubProcess, Activity, Event, Gateway
isPrecededBy	SubProcess, Activity, Event, Gateway	SubProcess, Activity, Event, Gateway
isPerformedBy	Activity	Actor
isExecutorOfActivity	Actor	Activity
usesInput	Activity	DataStore, DataObject
producesOutput	Activity, Event	DataStore, DataObject
hasExtendedAttribute	Activity, Gateway, Event	ExtendedAttribute
isAnnotatedBy	SubProcess, Activity, Event, Gateway	Annotation
exchangesMessageWith	SubProcess, Activity, Event, Gateway	SubProcess, Activity, Event, Gateway

The isExecutedAfterParallelExecutionOf object property defines the relation between a sub-process, activity, event, or gateway with the parallel flow that precedes it, if this occurs. The usage constraint of this property is related to one of the good practices that must be applied to the business process models used in the systematic (see Section 3.1). This restriction defines that all elements between two parallel gateways indicate the existence of a parallel flow. All

Table 4: Advanced object properties for the ontology.

Property	Domain	Range
isActivateWhenEventIsTriggered	SubProcess, Activity, Event, Gateway	Event
isExecutedWhen[Name of the activity before the gateway]OutputIs	SubProcess, Activity	GatewaysOutputs
isExecutedAfterParalelExecutionOf	SubProcess, Activity	ParallelFlow (accepts more than one)
isExecutedAfterInclusiveExecutionOf	SubProcess, Activity	InclusiveFlow (accepts more than one)

parallel flows that must be run before an activity are included as subclass of the ParallelFlow class and associated with the isExecutedAfterParallelExecutionOf property.

The isExecutedAfterInclusiveExecutionOf object property presents practically the same characteristics as the isExecutedAfterParallelExecutionOf property. The only difference is that its usage restriction is defined by two inclusive gateways. All elements between two inclusive gateways define an inclusive flow. All inclusive flows that must be run before an activity are included as a subclass of the InclusiveFlow class and associated with the isExecutedAfterInclusiveExecutionOf property.

4.2 Queries about the Generated Ontology

The PM2ONTO tool provides predefined queries, structured in the SPARQL language, for users to obtain information about the ontology. Thus, users do not need to manually construct a SPARQL query. Queries are classified into two groups: basic and advanced.

Basic queries provide information about BPMN elements, which have been mapped as classes to the ontology, regardless of the relations defined for those classes. These queries can use the following data properties defined for the classes as filters: identifier, name, description, and documentation. It is also possible to filter the SubProcess, Activity, Event, Gateway, and Artifact classes by the "type" attribute. The expected outputs for basic queries are the main information about one or more BPMN elements, i.e. identifier, name, type, description, and documentation. Queries about any element mapped to the ontology from the metamodel may contain filters related to the values of their data properties.

Advanced queries provide information about the relations and interdependencies between the classes generated for the BPMN elements. The object properties defined for the classes of the ontology enabled the definition of the advanced queries, which are listed below (PM2ONTO tool version):

1. Activities performed by a specific actor;
2. Predecessor/successor element of a filtered element in the query;
3. Elements that perform messages exchanges;
4. Elements identified as functional requirements;
5. Elements identified as non-functional requirements;
6. Elements identified as business rules;
7. Elements that use an artifact as input;
8. Activities that produce an artifact as output (Fig. 6);
9. Activities preceded by an exclusive gateway;
10. Activities preceded by an inclusive gateway;
11. Activities preceded by a parallel gateway.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX j.0: <http://www.semanticweb.org/ontology/exampleOntology#>
PREFIX j.1: <http://www.semanticweb.org/ontology/exampleOntology#producesOutput#>

SELECT DISTINCT ?id ?name ?description ?documentation
WHERE
 { ?x    rdfs:subClassOf    j.0:Activity ;
         j.0:id             ?id .
         j.0:name           ?name .
         j.0:description    ?description .
         j.0:documentation  ?documentation .
  ?activity rdfs:subClassOf ?activity_produces_output_restriction .
  ?activity_produces_output_restriction
         owl:onProperty  j.1:
 }
```

Figure 6: SPARQL code for query number 8.

5 EXAMPLE OF APPLICATION OF THE PROPOSED PROCESS

In order to show how to apply the systematized process presented in Section 3, a business process model in BPMN v2.0 was selected from the repository: http://www.bizagi.com/en/community/process-xchange. There are models with different levels of complexity in this repository, but all of them

have been elaborated with the Bizagi modeling system. This system allows the export of a business process model to XPDL v2.2 and also allows the use of extended attributes in the BPMN elements documentation.

The choice of model took into account that the conditions presented in Section 3.1 should be satisfactory. Additionally, the following characteristics were considered in order to better demonstrate the process: (a) textual documentation considered appropriate for all the BPMN elements of the model; (b) complexity level of the model considered from medium to high; (c) presence of sub-processes; (d) presence of extended attributes; (e) presence of more than one pool.

The "Accounts Payable" business process model satisfies the requirements of Section 3.1 and the requirements from (a) to (e), although it required some adjustments to supplement the documentation. This model refers to the processes of an Accounting Department, including submission of documents by suppliers and subsequent release of the payment after the validation of them. Fig. 7 shows the Accounts Payable model with its subprocess.

In order to apply the Step 1, Bizagi modeling system (v3.1.0.01) was used to export the model to XPDL v2.2 files. A folder with the XPDL file was then generated. This file was selected as the input for the PM2ONTO tool in order to apply the Step 2. The steps from 2.1 to 2.3 were then executed by the tool, generating the ontology of the Accounts Payable Model. Figs. 8 and 9 show excerpts from the ontology according to the visualization of the Protégé system (v5.2.0). In these excerpts, interdependencies of activities and sub-processes with other activities and sub-processes are evidenced by directing the result through an exclusive gateway.

The complete ontology presented the following results:

- 36 classes;
- 129 relations, axioms classes defined from the object properties;
- 4 business rules;
- 6 Functional Requirements;
- 3 Non Functional Requirements.

It should be noted that the business rules, the functional requirements and the non-functional requirements will also assist the Information Technology (IT) team. These elements can be obtained in the ontology from the extended attribute classes, which have the properties of boolean data (isBusinessRule, isFunctionalRequirement and isNonFunctionalRequirement) defined with the value "true". Thus, all business rules, functional requirements and non-functional requirements can be queried using the generated ontology.

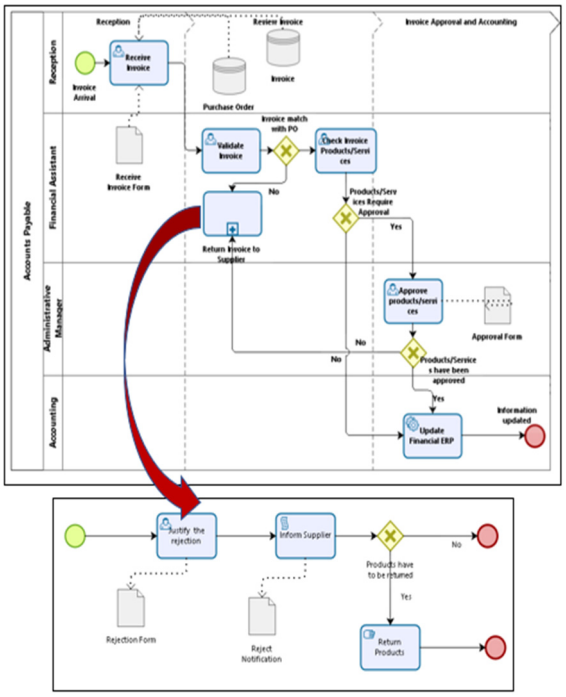

Figure 7: Accounts Payable Model from Bizagi Repository.

It is important to mention that all the BPMN elements and the associated documentation were mapped to the ontology as well as the relations between the element. The interdependencies were more explicit in the ontology than in relation to the business process model. The advanced object properties, present in the ontology, facilitated the understanding of some sequences of elements in the model, such as the property isExecutedWhen[Activity name]OutputIs (Figs. 8 e 9). This property was defined to improve understanding of all the business process model parts that are formed by a decision, indicated by an exclusive gateway. Moreover, since the interdependencies were defined as object properties, the queries about the BPMN elements from their

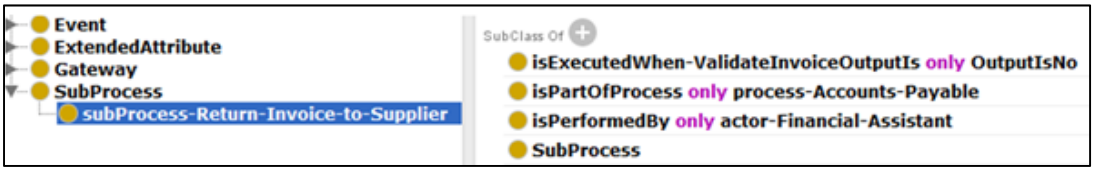

Figure 8: Relations obtained for a subprocess in the ontology.

Figure 9: Relations obtained for an activity in the ontology.

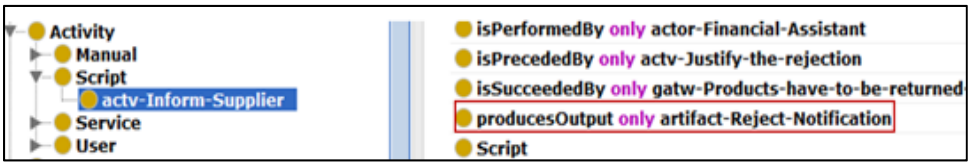

Figure 10: Result displayed in the Protégé system for the query exemplified as "8".

relations became possible. Thus, the queries mentioned in Section 4.2 can be made using the PM2ONTO tool. Table 5 shows the answers to some query examples for the "Accounts Payable" ontology, generated from the Accounts Payable Model. That table shows examples of filters (second column) that can be passed to the PM2ONTO tool for the 11 types of available queries (first column). The results shown in Table 5 are in textual format. However, the PM2ONTO tool allows results to be displayed in the OWL language. An example of such a case is shown in Fig. 10, which illustrates how the query output exemplified as "8" in Table 5 can be viewed automatically in the Protégé system.

Table 5: Examples of results for the predefined queries.

Nº	Filters	Results
1	Actor Name = 'Recepcionist'	Activity Name = 'Receive Invoice'
2	Activity Name = 'Validate Invoice'	Predecessor Name = 'Receive Invoice', Successor Name = 'Invoice match with PO'
3	Activity Name = 'Validate Invoice'	-
4	Extended Attribute Name = 'FR'	Extended Attribute Name = 'FR'
5	Extended Attribute Name = 'RULE'	Extended Attribute Name = 'RULE'
6	Extended Attribute Name = 'NFR'	Extended Attribute Name = 'NFR'
7	Activity Name = 'Receive Invoice'	Data Object Name = 'Receive Invoice Form'
8	Activity Name = 'Inform Supplier'	Data Object Name = 'Reject Notification'
9	Gateway Name = 'Invoice match with PO'	Activity Name = 'Validate Invoice'
10	Gateway Name = 'Invoice match with PO'	-
11	Gateway Name = 'Invoice match with PO'	-

6 CONCLUSION

The goal of this paper was to present a process to map elements and knowledge extracted from a business process model in BPMN to an ontology. The PM2ONTO tool was introduced to support the process of automatic generation of the ontology and provide means for a human user to consult information about the business process model. The paper showed how the mapping was designed and implemented in the PM2ONTO tool. The application

of the process for generating the ontology with the support of the PM2ONTO tool was exemplified using a real model in this paper.

In order to validate the ontology, criteria based on the works of Fanesi et al. (2015) and Pizzoleto and Oliveira (2017) are being elaborated. For this, organized questions about various aspects of the business processes model are key tools.

The alternative way of representing a business process model using an ontology brings several benefits. Whereas ontologies generated are readable structures and manipulated by machine, they allow the application of various techniques and interactions with digital systems.

The navigation of the model through the ontology allows better understanding of the activities and resources, complementing the knowledge about the business process model. Knowledge can be embedded in the ontology and shared between business teams and other interested teams, such as the IT teams. This helps to approximate the views of business and IT teams and, consequently, assists the processes of Requirements Engineering for software being developed for the organization.

Using ontology integration techniques, business process models can be integrated, providing broader queries of interest to the organization. Text mining techniques can be applied to ontologies, allowing to improve the system of strategic queries. Information visualization techniques can also be applied to business process models using their ontological representations.

REFERENCES

Apache Software Foundation. (2017), "Apache Jena – a free and open source java framework for building Semantic Web and Linked Data applications", avaliable at: https://jena.apache.org.

BOC Group. (2013), "Business Process Management with Adonis", avaliable at: https://uk.boc-group.com/adonis.

Correia, A. and Abreu, F. B. (2015), "Enhancing the correct-ness of BPMN models", in Varajão, J. E. Cruz-Cunha, M. M. and Martinho, R. (Eds.), *Improving organiza-tional effectiveness with Enterprise Information Systems*, Hershey: IGI Global, pp. 241-261.

Fanesi, D., Cacciagrano, D. R. and Hinkelmann, K. (2015), "Semantic Business Process Representation to Enhance the Degree of BPM Mechanization - An Ontology", *Proceedings of IEEE International Conference On Enterprise Systems*, Basel, pp. 21-32.

Gábor, A. and Kö, A. (2016), "Corporate Knowledge Discovery and Organizational Learning: The Role, Importance, and Application of Semantic Business Process Management", *Knowledge Management and Organizational Learning*, Springer International Publishing.

Gómez-Pérez, J., M. (2010), *Studies on the Semantic Web: Acquisition and Understanding of Process Knowledge Using Problem Solving* Methods, IOS Press, Amsterdam.

Guido, A., Pandurino, A. and Paiano, R. (2016), "An Ontological Meta-Model for Business Process Model and Notation", *International Journal of Business Research and Management (IJBRM)*, Vol. 7, pp. 1-13.

Haller, A., Marmolowski, M., Oren, E. and Gaaloul, W. (2007), "oXPDL: a Process Model Exchange Ontology", *DERI–Digital Enterprise Research Institute*.

Missikoff, M., Proietti, M. and Smith, F. (2010), "Linking ontologies to business process schemas", *IASI-CNR Tech*, Vol. 20, pp. 1-20.

Nogueira, F. A. and Oliveira, H. C. (2017), "Application of heuristics in business process models to support software requirements specification", *Proceedings of the 19th International Conference on Enterprise Information Systems*, Funchal, pp. 1-12.

Pizzoleto, A. V. and Oliveira, H. C. (2016), "A systematic approach to evaluate enterprise ontologies using testing techniques of software usability". *Proceedings of the 14th International Conference on Software Engineering Research and Practice*, Las Vegas, pp. 125-131.

Przybylek, A. (2014), "A business-oriented approach to requirements elicitation", *Proceedings of the 9th International Conference on Evaluation of Novel Approaches to Software Engineering*, Lisbon, pp. 1-12.

Ternai, K., Török, M. and Varga, K. (2016), "Corporate Semantic Business Process Management", in Gábor, A. and Kõ, A. (Eds.), *Corporate Knowledge Discovery and Organizational Learning: The Role, Importance, and Application of Semantic Business Process Management*, Springer Internationl Publishing, pp. 33-57.