BBO: BPMN 2.0 Based Ontology for Business Process Representation

Amina Annane, Nathalie Aussenac-Gilles, Mouna Kamel
IRIT, CNRS, University of Toulouse, Toulouse, France
amina.annane@irit.fr
nathalie.aussenac-gilles@irit.fr
mouna.kamel@irit.fr

Abstract: Any industrial company has its own business processes, which is a number of related tasks that have to be executed to reach well-defined goals. In order to analyze, improve, simulate and automate these processes, it is essential to represent them in a formal way. The activity of representing business processes is known as Business Process Modelling (BPM); it is an active research area that attracts more and more attention with the emergence of Industry 4.0. Semantic Web technologies, especially ontologies, are promising means to advance BPM and to realize the Industry 4.0 vision. In this scope, we developed the BBO (BPMN 2.0 Based Ontology) ontology for business process representation, by reusing existing ontologies and meta-models like BPMN 2.0, the state-of-the-art meta-model for business process representation. We evaluated BBO using schema metrics, which showed that it was a deep and rich ontology with a variety of relationships. Thanks to a use case, we illustrated the ability of BBO to represent real business processes in a fine-grained way and to express and answer the competency questions identified at the specification stage.

Keywords: BPMN 2.0, business process modeling, ontology, Industry 4.0, Semantic Web

1. Introduction

Business processes (BPs) are a key knowledge in any industrial company. Indeed, they are the procedures that describe the required activities to produce commercial products: "*a process is a particular procedure for doing something involving one or more steps or operations. The process may produce a product, a property of a product, or an aspect of a product*" (ISO, 1998). To analyze, simulate, improve and automate these processes, it is essential to formally represent them (Rospocher, Ghidini and Serafini, 2014). The activity of representing BP is known as Business Process Modelling (BPM), which becomes more and more important in Industry 4.0 era. Indeed, the Industry 4.0 vision assumes an effective and high-quality communication between systems (interoperability), and between humans and systems. Semantic web technologies are promising solutions to realize this vision (Vogel-Heuser and Hess, 2016). Ontologies are the key element for representing data in structured way on the semantic web (Berners-Lee *et al.*, 2001). They provide a formal representation, and ensure interoperability and communication between different systems. Moreover, thanks to their reasoning ability, ontologies contribute to check data integrity and consistency (Rospocher *et al.*, 2014; Roy *et al.,* 2018). Hence, representing BPs using ontologies seems to be a solution to overcome the interoperability challenge. In the AVIREX project, we investigate the development of a **generic** ontology that should enable a fine-grained representation of industrial BPs. The first two use contexts are given by the project partner industrial companies, Thales Alenia Space (TAS) and Continental. In both cases, the BPs define how to monitor and supervise the automatic or manual assembly of electronic, digital and physical components, of a car equipment in Continental or of a space vehicle equipment bay in TAS. Once populated with BP data, the ontology forms a knowledge base (KB) that will be exploited by a virtual agent to support operators in the execution of BP step-by-step. It will also provide answers to operators' questions about the process execution.
Modeling BP is a very well researched subject and a very complicated one. We can benefit from previous formalization initiatives by reusing existing ontologies and models rather than developing a new ontology from scratch. In the literature, Business Process Model and Notation (BPMN) is the most adopted meta-model for representing BPs (OMG., 2011). Indeed, it is a standard for BPM maintained by the Object Management Group (OMG). In spite of its industrial maturity, BPMN does not support the representation of some process specifications such as the material resources required to carry out a given task, or the workstation where a given task should be performed. These specifications are essential for a complete description of BPs as stated by (Falbo and Bertollo, 2009): "*A process should be defined considering: the activities to be accomplished, the required resources, the input and output artefacts, the adopted procedures (methods, techniques, templates and so on) and the life cycle model to be used*."
In this paper, we report how we have built and evaluated a BPMN based ontology that we called BBO. The core of BBO is an ontological representation of a fragment extracted from the BPMN 2.0 meta-model. Hence, we exploited the process-execution specifications of BPMN 2.0. In addition, we have extended the core of BBO with taxonomies, concepts, relations and attributes to meet the specifications presented in Section 3. Related

works are reported in Section 2, as well as a presentation of METHONTOLOGY (Fernandez et al., 1997), the methodology followed to develop BBO in five classical stages. *Specification* states why the ontology is being built, its intended uses and who are the end-users (Section 3). *Conceptualization* consists in structuring the domain knowledge in a conceptual model using the terms identified in the specification phase (Section 4). *Formalization* transforms the conceptual model into a formal model using a knowledge representation language. *Implementation* builds computable models in a computational language. BBO *Formalization* and *Implementation* are presented in Section 5*. Maintenance* is dedicated to ontology updates and corrections once it has been deployed, which is not yet the case for BBO. We have evaluated BBO with schema metrics and use cases of the AVIREX industrial collaborators (Section 6). Thanks to a use case, we show the BBO ability to represent BPs in a fine-grained way, and to answer competency questions. Finally, we conclude the paper.

2. Related work

Representing BPs was the focus of several research projects such as Enterprise (Uschold *et al.*, 1998) and SUPER (Semantics Utilised for Process management within and between EnteRprises) (Hepp and Roman, 2007). Many of these projects proposed BP models of which we studied those that best met the specifications described in Section 3. Event Process driven Chain (EPC) (Scheer e*t al.,* 2005) and BPMN are the most known BP meta-models (and graphical languages). However, BPMN offers a finer grained representation than EPC and an execution logic for its elements. BP ontologies have often overlapping fragments and are either very general (Uschold *et al.*, 1998; Van Grondelle and Gülpers, 2011; Abdalla *et al.*, 2014), or specific to a given type of processes: (Ru´IZ *et al.*, 2004) propose an ontology for software project management; (Falbo and Bertollo, 2009) also present a software process ontology; (Karray, Chebel-Morello and Zerhouni, 2012) describe an ontology for industrial maintenance processes; (Chungoora *et al.*, 2013) report an ontology for the manufacturing process. None of these ontologies meets all the specifications presented in Section 3, but some of them include relevant fragments that we have reused such as BPMN or the resource taxonomy (Section 4).
The idea of converting BPMN into an ontological model has been investigated in two previous works. In (Rospocher, Ghidini and Serafini, 2014), BPMN 1.0 is transformed into an ontology that has been manually revised and enriched with annotations and axioms. A more recent and richer version, BPMN 2.0, has been published in 2011. It contains a full formalization of the execution semantics for all BPMN elements. Hence, it is worth using BPMN 2.0. Natschläger (2011) has proposed an ontological version of BPMN 2.0. However, to the best of our knowledge, this ontology is not available for the community. Another ontology (BPMN-onto, 2019) has been automatically extracted from BPMN 2.0, but we were not able to find any documentation about how it was generated. Moreover, this ontology contains no annotations and much less information than the specification document. In all these works, the idea was to transform the whole BPMN meta-model into an ontology. This was not our goal. Instead, we extracted a fragment from BPMN that deals with process description, and then we extended this fragment to satisfy BBO specifications. In spite of the industrial maturity of BPMN, these extensions are required as it has already been stated in the literature (Awad *et al.*, 2009; Marcinkowski and Kuciapski, 2012; Bocciarelli *et al.*, 2016).

3. Specification

Our ontology will be exploited only by a virtual assistant to monitor process execution step by step, and to answer questions about processes. To determine the scope of the ontology, we have exploited two knowledge sources: (i) technical documents describing industrial BPs, and (ii) competency questions that we have collected from related works and interviews with operators (who perform the task) and experts ((business analysts that manage business process models).

3.1 Technical documents

In the AVIREX project, we collaborate with two industrial companies, namely TAS and Continental. They have provided us with 20 technical documents that describe their BPs. Each document is between 10 and 30 pages long. It describes all the stages, devices and resources required to perform one BP. In general, a BP is organized in a set of sub-processes. A sub-process, in turn, may be decomposed into a set of sub-processes or tasks presented as separate sections or as vertical item-lists in the document. Tasks are the work to be performed by operators. Figure 1 shows a fragment of a real industrial BP with three tasks (a, b, c).

On the MMA station :

a. Execute the script MUXF.exe
b. Using the software SOFT, verify whether the script execution has generated an alarm
c. If the execution has generated an alarm, execute the process of solving alarm issues

Figure 1. Fragment of industrial business process description. Highlighted named entities are anonymized.

3.2 Competency questions

Competency questions are recognized to be a good means to materialize ontology specifications (Grüninger and Fox, 1995). After meeting experts and analyzing related works (Falbo and Bertollo, 2009; Abdalla *et al.*, 2014), we collected a set of 22 competency questions from which we give a sample in Table 1.

Table 1. Example of competency questions

• Which resources are required by an activity?	• Which resources are produced by a given activity?
• Which sub-activities is an activity decomposed?	• What is the type of a resource?
• Which activities must precede a given activity?	• Who should perform a given activity?
• Where the activity should be performed?	• ….

3.3 Synthesis

Based on the study of the previous knowledge sources, we have identified five main concepts that must be covered by BBO ontology:

1. **Process**: it is the key concept. The ontology should enable (i) to represent the decomposition of a given process in activities (sub-processes and tasks), and (ii) to well describe the order in which these activities should be performed, which may be controlled by events and conditions.
2. **Input/output specifications**: tasks may require some input requirements (resources or parameter values) before being performed, or they may produce outcomes (resources or parameter values). As we can see in Figure 1, resources are of different types (e.g., files, software, etc.).
3. **Agent**: the actor that performs a given process activity. Indeed, it is important to specify who is responsible for the accomplishment of a given activity.
4. **Work product**: to specify the process or processes that are required to produce a given product. Moreover, it is necessary to represent the composition of products.
5. **Manufacturing facility**: the place where the process activities should be performed.

The complete list of competency questions grouped by key concepts is available on the following link:
https://github.com/AminaANNANE/BBO_BPMNbasedOntology/blob/master/Competency%20questions.txt

4. Conceptualization

In the following, we present the conceptual model of BBO using UML class diagrams. We use different colors for UML classes: green boxes are BPMN classes, blue boxes are reused classes from existing ontologies, and orange boxes are new classes that we added to BBO. For the sake of presentation, we split BBO model into five fragments according to the five main concepts identified in the Specification stage.

4.1 Process

BPMN 2.0 (OMG., 2011) is a state-of-the-art meta-model for BP representation. BPMN 2.0 has been developed for several years by experts from OMG team and industrial collaborators. Given the industrial maturity of BPMN (used for BPMN 2.0 from now on) meta-model, and its well-defined execution semantics, it is worth reusing it. We manually studied the BPMN 2.0 specification document (OMG., 2011), more than 500 pages, to select the fragments dedicated to the representation of processes. Indeed, BPMN meta-model is richer than need to meet BBO specifications. In particular, we left aside the classes and relations required to graphically represent elements or interactions between processes (i.e., collaboration, choreography, conversations, etc.). Figure 2 shows the main BPMN concepts that we reuse. *Process* is a sub-class of *FlowElementsContainer*. Describing a process consists in defining the *FlowElements* that compose it. *FlowElements* class has two sub-classes: *SequenceFlow* and *FlowNode. SequenceFlow* represents transitions that ensure the move from the source *FlowNode* to the target one. A *SequenceFlow* may depend on a given condition, which is represented as an instance of *Expression* class. *FlowNode* class groups the activities that compose a process:

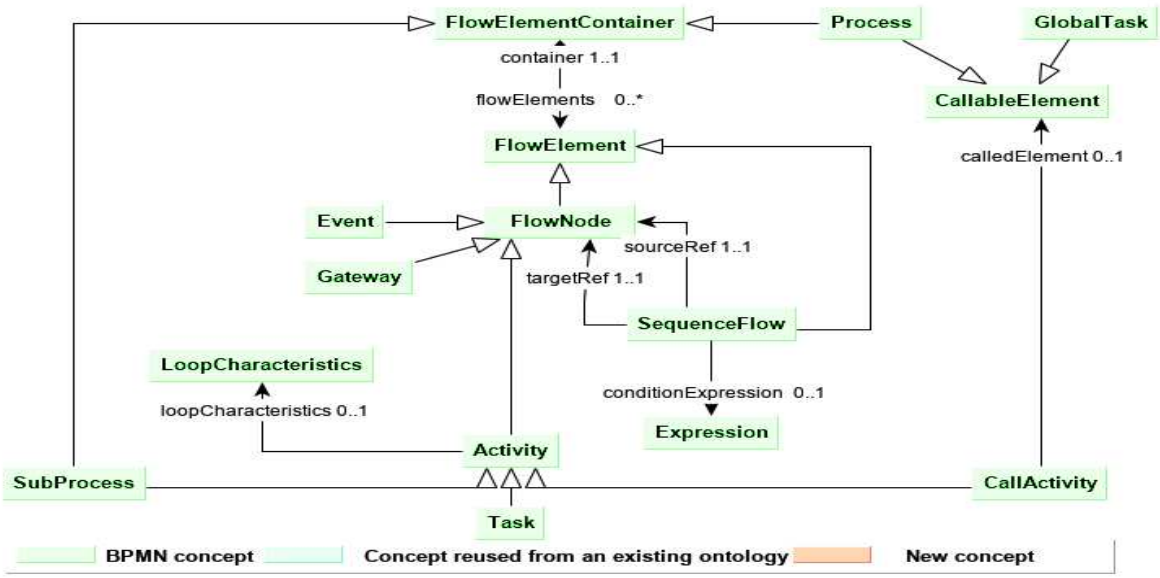

Figure 2. *Process* class properties and related concepts from BPMN reused in BBO.

- *Activity* is the work to be performed. Activity class has three sub-classes :
  a. *Task* : an atomic task
  b. *Sub-Process*: complex task that contains several *Task*s
  c. *CallActivity:* an activity that calls a *CallableElement* that may be a *GlobalTask (i.e.,* a reusable task) *or Sub-Process*.

  Activity class is related to LoopCharacteristics to represent iteration specifications.
- *Event* is something that "happens" during the course of a process. Events affect the flow of the process and usually have a cause or an impact and may require or allow for a reaction. In BPMN, events may be interrupting (i.e, when the events occurs, the activity related to this event is stopped) or not. Moreover, several types of event exist: TimerEvent, ConditionalEvent, etc.
- *Gateway* is used to control how *SequenceFlows* interact as they converge or diverge within a *Process*.

Most BPMN specifications (OMG., 2011) are available as UML diagrams. However, many specifications are only expressed in natural language, from which we extracted new concepts (see Table 2) to be inserted in BBO. Let us take an example to illustrate the process. In BPMN meta-model, Gateway class has an attribute that determines the gateway type: "converging", "diverging", "multiple" or "unspecified". Gateways that have the value "converging" are defined as follows: "A *Gateway* with a gateway Direction of converging **MUST** have multiple incoming *Sequence Flows*, but **MUST NOT** have multiple outgoing *Sequence Flows*" (p. 290) (Fig. 3). The simple assignment of "Converging" value to a given *Gateway* instance does not guarantee the coherence with this definition. Therefore, we explicitly represented the four types as four sub-classes of the *Gateway* class and formalized their definitions thanks to cardinality restrictions (see § 4.1.2 for formalization details).

Table 2. Additional classes according to textual BPMN specification

Concept	**Sub-concepts**
SequenceFlow	ConditionalSequenceFlow, DefaultSequenceFlow, NormalSequenceFlow
SubProcess	EventBasedSubProcess
Gateway	ConvergingGateway, DivergingGateway, MixedGateway, UnspecifiedGateway
EventBasedGateway	ExclusiveEventBasedGateway, parallelEventBasedGateway
Event	cancelEvent, conditionalEvent, ErrorEvent, MultipleEvent, NoneEvent, TimerEvent, etc.
Expression	UnderspecifiedExpression

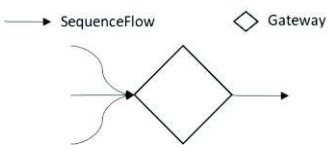

Figure 3. Graphical representation of a converging gateway.

BPMN meta-model covers well all the aspects related to the *Process* concept: decomposition of a process into sub-activities, the temporal and conditional constraints between activities, etc. However, it does not fully support the other four main concepts presented in Section 3. For instance, with BPMN meta-model it is not possible to specify that the task (b) in Figure 1 requires a resource SOFT of type Software, or that the three tasks should be performed on the "MMA station" (MMA station is a test bed for satellites). Consequently, it was necessary to enrich the BPMN fragment with more concepts and relations to satisfy BBO specifications.

4.2 Input/output specifications

As explained earlier, we need to represent the input and output of each activity. In BPMN meta-model, an activity may have at most one *InputOutputSpecification* that is related to the required Input/Output Data. However, we need to specify Input/Output of various types, not only those of data type. Therefore, we have added the two relations "has_resourceInput" and "has_resourceOutput" between *InputOutputSpecification* class and *Resource* class (Figure 4). Moreover, to represent the parameter values specifications, we have added the concepts: *ParameterValueBinding*, *Parameter* and its subclasses *QualitativeParameter* and *QuantitativeParameter*, *ParameterValue*, *ParameterExpectedValue*, and *UnitOfMeasure*. The *UnitOfMeasure* class is specified using the two concepts *Unit* and *Prefix* of the unit measures ontology UO (UO-onto, 2019).
The *Resource* concept exists in the BPMN meta-model. However, its semantics and definition are ambiguous. Indeed, on p. 95 of BPMN specification, the Resource class is supposed to cover all resource types. However, the definition of the relation that assigns resources to a process (p. 148) or an activity (p.152), limits the set of resources to the agents responsible for performing the work. The last definition seems to be most adopted.

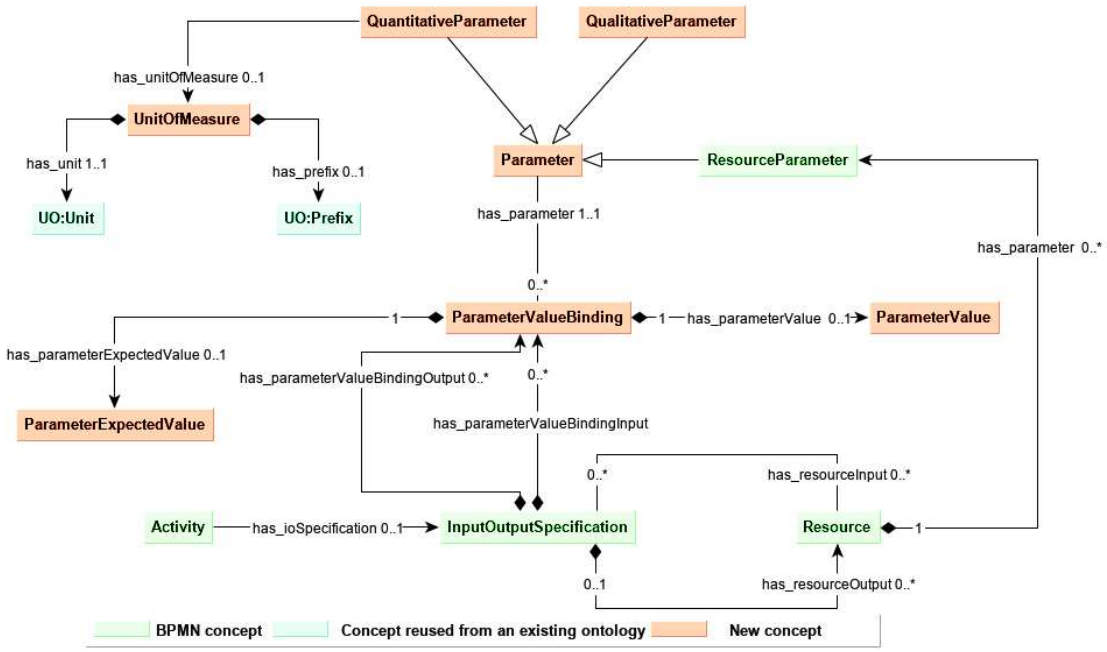

Figure 4. *InputOutputSpecification* class, linked properties and classes in BBO.

Indeed, in (Awad *et al.*, 2009; Stroppi, Chiotti and Villarreal, 2011) *Resource* in BPMN is equivalent to *Agents*. In BBO, like in (Karray *et al*., 2012), we adopt the first definition of *Resource*, that englobes all resource types. Hence, we may define a resource taxonomy (Figure 6). This taxonomy actually is relevant to answer to some competency questions like "What is the type of a given resource?" BBO resource taxonomy is inspired from similar ones proposed in (Falbo and Bertollo, 2009; Karray, Chebel-Morello and Zerhouni, 2012).

4.3 Manufacturing facility

To specify where the task should be performed, we reused the taxonomies introduced in (Chungoora *et al.*, 2013) and (Fraga, Vegetti and Leone, 2018) and obtained the part of the ontology shown in Figure 5. A workstation, *Station*, is where a particular job is performed. *Cell* is the place that groups a set of related operations in the production flow, while *Shop* is the area where production is carried out, and *Factory* is the place where those production areas are located. We limit *Task* to have at most one *ManufacturingFacility*,

while other types of activities – complex ones with several tasks like *Process* – may require several manufacturing facilities. Indeed, a *Task* is an atomic unit of work and should be performed at one place.

Figure 5. ManufacturingFacility class and its sub-classes in BBO.

4.4 Work product

In (ISO, 2005), the term *Product* is defined as follows "*Thing or substance produced by a natural or artificial process*." We adopt this definition for the *WorkProduct* concept. In addition, we consider *WorkProduct* as a particular type of *MaterialResource*, as states the ISO definition: "*Resource is the result of a process."* ISO 10303-239. Indeed, once produced, the product may be considered as resource and used as an input for another activity. To define the composition of a given product, we added the relation "is_composedOf" between *WorkProduct* and *Resource* (Figure 6).

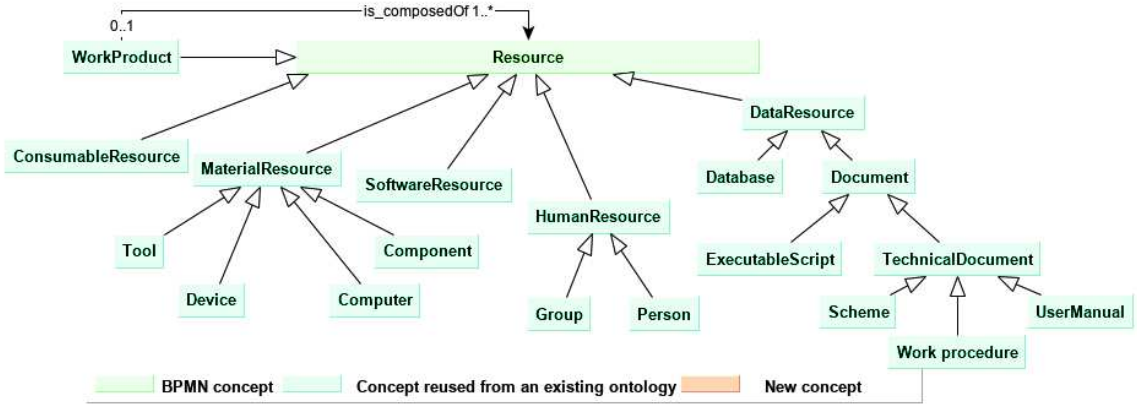

Figure 6. BBO resource taxonomy.

4.5 Agent

We reused the *Agent* sub-ontology proposed in (Ru´IZ *et al.*, 2004), and presented in Figure 7.

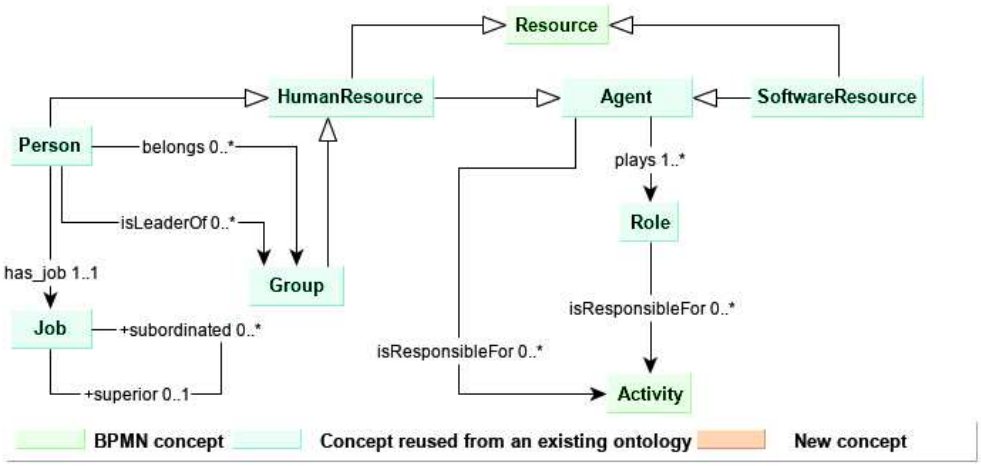

Figure 7. Agent class with linked properties and classes in BBO.

An *Agent* may be a *HumanResource* or a *SoftwareResource*. The concept *Job* with the two relations "subordinated" and "superior" represent the organizational model of the company, which is not supported by

BPMN meta-model. Note that, we differentiated *Job* from *Role* to offer more flexibility. Indeed, two persons that have the same *Job*, may have different authorization levels to execute *Activities.* For a given *Activity*, we may assign a specific *Agent* (i.e., direct assignment), or a *Role* (i.e., indirect assignment). In the case of indirect assignment, all agents playing the assigned role are potential performers of the *Activity*.

5. Formalization and Implementation

We have formalized and implemented the conceptual model of BBO in OWL 2 DL using Protégé. First, we designed and applied a set of conversion rules that automatically generated an OWL representation from the UML diagrams of BBO. Second, we manually turned various natural language specifications in the BPMN document into a formal OWL representation.

5.1 Formalizing UML class diagrams in OWL

Generating an OWL representation from the UML diagrams results in the following algorithm:

- For each UML class, create an owl class
- For each relation between UML classes create an OWL ObjectProperty
- For each class attribute: If the type is an UML class, create an OWL ObjectProperty. Otherwise, create an OWL DataProperty
- Cardinalities have been transformed into qualified cardinality restrictions. Let UClass1, UClass2 be two UML classes, and OClass1, OClass2 their corresponding OWL classes; UProperty be an UML relationship or attribute, and OProperty its corresponding OWL property; x be an integer different than 0 or n:
    - UClass1 UProperty [x..x] UClass2 => OClass1 **subClassOf** OProperty **exactly** x OClass2
    - UClass1 UProperty [x..n] UClass2 => OClass1 **subClassO**f OProperty **min** x OClass2
    - UClass1 UProperty [0..x] UClass2 => OClass1 **subClassOf** OProperty **max** x OClass2

In the BPMN-Process class diagram, relationships with the same name are used several times between different classes. In contrast, in an ontology the names (identifiers) of object and data properties must be distinct if they have a different semantics. If they have the same meaning, then they must be restrictions of the same more general property, which means that the domain and range of the property are super-classes of all the classes linked by this property in the meta-model.

5.2 Formalizing BPMN natural language specifications in OWL

BPMN 2.0 specification provides a meta-model for BPMN elements as a UML class diagram and in the form of an XML schema. However, the diagrams and XML schema do not reflect the whole specification and miss a part of its semantics (Dijkman, Dumas and Ouyang, 2008; Wong and Gibbons, 2008). Consequently, even a formal translation of the UML and XML specifications would not enable checking the consistency of represented processes, because a large part of the specifications is in natural language. Therefore, we have tried to manually conceptualize the natural language specifications, and formalize them in OWL.

Table 3. Examples of conversions from natural language specifications to restrictions on properties.

Specification in natural language	Formalized Specification
A *Start Event* **MUST** be a source for a Sequence Flow. p.245	StartEvent **subClassOf** has_outgoing **some** SequenceFlow
End Event ends the flow of the *Process*, and thus, will not have any outgoing Sequence Flows. pp 246.	EndEvent **subClassOf not** (has_outgoing **some** SequenceFlow)
List of BPMN elements that **MUST NOT** be used in an Ad-HocSub-Process : Start Event, End Event. p.182	AdHocSubProcess **SubClassOf not** (has_flowElements **some** (StartEvent or EndEvent))

These specifications may be classified into two categories: (i) Specifications that lead to define constraints on existing properties, and (ii) Specifications that lead to define new classes as restrictions on existing classes. Table 3 and 4 show some examples of these categories, and the corresponding formal representations defined from each sentence. As a first stage, for each BPMN element, we added its description as mentioned in BPMN 2.0 specification (OMG., 2011) using the rdfs:comment property. Class definitions in Table 4 are particularly interesting for the automatic classification of instances. According to the various reasoners in Protégé (i.e., Hermit, Fact and Pellet), the ontology is consistent and remains consistent, even after its population with assertions describing several BP models.

Table 4. Examples of conversions from natural language specifications to new class definitions

Specification in natural language	Formalized Specification in OWL
A *Gateway* with a gatewayDirection of converging **MUST** have multiple incoming *Sequence Flows*, but **MUST NOT** have multiple outgoing *Sequence Flows.* p. 290	ConvergingGateway **equivalentTo** (Gateway **and** (has_incoming **min** 2 SequenceFlow) and (has_outgoing **exactly** 1 SequenceFlow))
Conditional SequenceFlow is a *SequenceFlow* that has a specified condition Expression. p. 97	ConditionalSequenceFlow **equivalentTo** (SequenceFlow **and** has_conditionExpression **some** Expression)
A Timer *Event* is an *Event* that has exactly one *TimerEventDefinition*. p. 274	TimerEvent **equivalentTo** (Event **and** (has_eventDefinition **exactly** 1 TimerEventDefinition))

6. Evaluation

We evaluated BBO using schema metrics and competency questions that we collected in Section 3.

6.1 Schema metrics

We used the two schema metrics introduced in (Tartir and Arpinar, 2007) to evaluate BBO: the relationship diversity (RD) and the schema deepness (SD). Let NR be the number of non-inheritance relationships, NH the number of inheritance relationships (i.e., isA) and NC the number of classes.

- RD = NR / (NR+ NH), which exceeds for us 50% (Table 5). This ratio indicates that BBO is not just a hierarchy of subclasses, but it is also rich with relationships that describe the knowledge domain.
- SD = NH/NC. This measure describes the distribution of classes across different levels of the ontology class hierarchy. The SD value of BBO (Table 5) is low (less than one hypernymy link per class): it means that the ontology is deep (or vertical): it covers a knowledge domain (i.e., BPs) in a detailed manner.

Table 3. BBO Schema metrics

Concepts	Relationships others than isA	isA relations	Metrics
106	125	83	• RD = 125/(125+83) = 0.60 • SD = 83/106 = 0.78

6.2 Business evaluation

We evaluate the ability of BBO to represent BPs described in the companies' technical documents, and to answer the competency questions in Section 3. For each BP description, (1) we represent this BP with BPMN graphical elements using an open source software, Camunda (https://camunda.com/): this step is not mandatory to instantiate BBO, however it is more convenient to communicate with experts and to validate BP representations; (2) we populate the ontology taking into account both the BPMN textual description and the graphical representation; (3) we design SPARQL queries corresponding to the competency questions and check their results. In the following, we will carry out all three steps on the real example presented in Figure 1. The evaluation of a larger process would require a complete report.

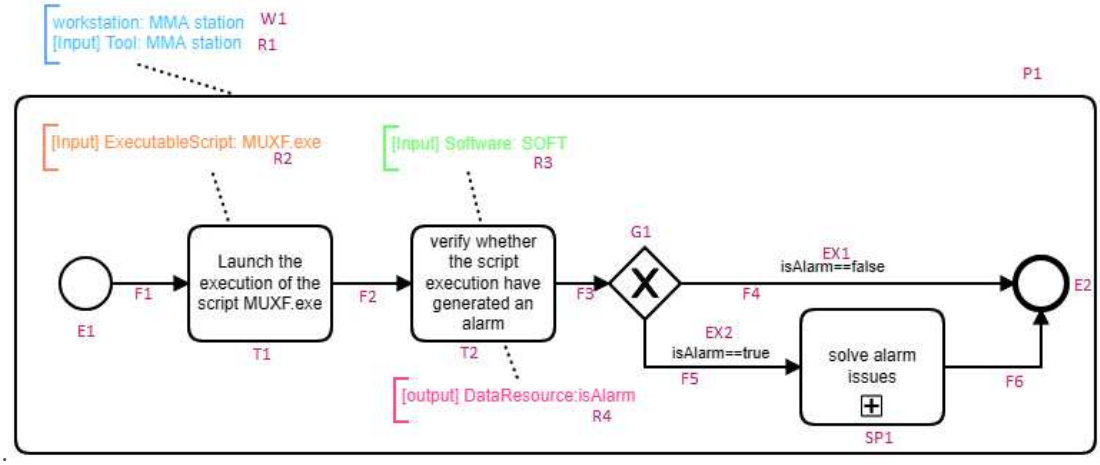

Figure 8. The representation of the example of Figure 1 with BPMN graphical elements.

Figure 8 shows the graphical representation of the Figure 1 example with BPMN graphical elements. The first circle and the last one denote the events that respectively start and end the process. Each rectangle

corresponds to a task. Then, using an exclusive gateway (i.e., the diamond form), we check the Boolean variable "isA*larm":* if "true", a SubProcess (SP1) should be performed to solve the alarm issue. We populated BBO with assertions representing the above process, which results in the set of triples listed in Table 6. The table does not report the data property assertions, neither does it expand the process "solve alarm issues" to keep this use case easily readable. Identifiers of instances are the one defined in Figure 8.

Table 4. Instantiation assertions

Class Assertions	ConditionalExpression (EX1)	has_resourceInput (IO1,R2)	has_targetRef (F5,SP1)
Process (P1)	ConditionalExpression (EX2)	has_resourceInput (IO2,R3)	has_targetRef (F6,E2)
StartEvent (E1)	InputOutputSpecification (IO1)	has_sourceRef (F1,E1)	has_conditionExpression (F4,EX1)
Task (T_i) i=1,2	InputOutputSpecification (IO2)	has_sourceRef (F2,T1)	has_conditionExpression (F5,EX2)
EndEvent (E2)	InputOutputSpecification (IO3)	has_sourceRef (F3,T2)	has_flowElements (P1,F_j) j=1,..,6
SequenceFlow (F_j)j=1,..,6	***Property assertions***	has_sourceRef (F4,G1)	has_flowElements (P1,T_i) i=1,2
Software (R3)	has_ioSpecification (T1,IO1)	has_sourceRef (F5,G1)	has_flowElements (P1,G1)
ExecutableScript (R2)	has_ioSpecification (T2,IO2)	has_sourceRef (F6,SP1)	has_flowElements (P1,E1)
Workstation (W1)	has_ioSpecification (P1,IO3)	has_targetRef (F1,T1)	has_flowElements (P1,E2)
Tool (R1)	takesPlaceAt (P1,W1)	has_targetRef (F2,T2)	has_flowElements (P1,SP1)
Gateway (G1)	has_resourceInput (IO3,R1)	has_targetRef (F3,G1)	
DataResource (R4)	has_resourceOutput (IO2,R4)	has_inputValue (EX2,R4)	
SubProcess (SP1)	has_inputValue (EX1,R4)	has_targetRef (F4,E2)	

Table 7 shows examples of how we turned competency questions into SPARQL queries. Even if the minimum cardinality of the property linking *Task* and *ManufacturingFacility* is one, the KB is still consistent because of the open world reasoning assumption. This problem is solvable by closed world reasoning, which is supported by reasoners such as Pellet. Another solution would be to consider the *ManufacturingFacility* of the *Process* that includes the Task. In our example, the process P1 takesPlaceAt W1. We had a correct answer for Question 3, but it is not always simple to answer to this question, because of the dynamic aspect of BPs. Indeed, for T2, the answer may only be known during the execution since it depends on condition evaluation. We have performed queries on the inferred ontology, which explains the three answers to question 4. A software developer may keep only the lowest type in the hierarchy as resource type.

Table 5. Querying the knowledge base (i.e., BBO + instantiation assertions).

PREFIX BBO: <http://BPMNbasedOntology#>

N°	**Competency question**	**Query in SPARQL**	**Query answer /***Comment*
1	What are the input resources of T1?	SELECT ?resource WHERE { BBO:T1 BBO:has_ioSpecification ?io. ?io BBO:has_resourceInputs ?resource.}	**R2**
2	Where should T1 take place?	SELECT ?place WHERE { BBO:T1 BBO:takesPlaceAt ?place. ?place a BBO:ManufacturingFacility}	*The answer is empty because we assigned no manufacturing facility to task T1.*
3	What is the next task after T1?	SELECT ?nextTask WHERE { BBO:T1 BBO:has_outgoing ?SequenceFlow. ?SequenceFlow BBO:has_targetRef ?nextTask. ?nextTask a BBO:Task}	**T2**
4	What is SOFT?	SELECT DISTINCT ?resource ?typeResource WHERE {?resource BBO:name "SOFT"^^xsd:string. ?resource a BBO:Resource. ?resource a ?typeResource.}	**SoftwareResource** **Resource** **NamedIndividual**

7. Conclusion

BPM is a challenging research field, especially in Industry 4.0 era. In this article, we presented BBO, an ontology to represent BPs. To develop this ontology, we reused fragments from existing meta-models and ontologies, in particular BPMN_2.0, state-of-the-art meta-model for BP representation. We implemented BBO in OWL and made it available online (https://www.irit.fr/recherches/MELODI/ontologies/BBO). Up to now, early and partial evaluations showed the strengths of BBO: (i) it is not just a taxonomy, but rather a rich model with a diversity of relationships; (ii) it is a deep ontology covering a specific knowledge domain with details; (iii) it is consistent; (iv) it provides the vocabulary required to answer all the competency questions collected from industrial experts and from the literature. The current version of BBO proposes the classes required to model a process run but it keeps no trace of previous runs. We plan to extend BBO so that it supports this. Moreover,

we need to systematically instantiate BBO with all the process descriptions in the companies' technical documents. We will implement a Natural Language Processing pipeline to carry out this task.

Acknowledgement

The AVIREX project is funded by a grant from Occitanie region, the FEDER-FSE Midi-Pyrénées and Garonne 2014-2020 program under the label 2017-AVIREX-IRIT-READYNOV INDUSTRIE DU FUTUR. The authors would like to thank their industrial partners, namely SimSoft-Industry, Thales Alenia Space and Continental.

References

Abdalla, A., Hu, Y.,Carral, D., Li, N., Janowicz, K. (2014) 'An Ontology Design Pattern for Activity Reasoning', in 5th Workshop on Ontology and Semantic Web Patterns (WOP2014). Riva del Garda, Italy, pp. 78--81.

Awad, A., Grosskopf, A., Meyer, A., Weske, M. (2009) Enabling resource assignment constraints in BPMN, Technical Report. Hasso Plattner Institute.

Berners-Lee, T., Hendler, J. and Lassila, O. (2001) 'The Semantic Web', Scientific American, 284(5), pp. 28--37.

Bocciarelli, P. et al. (2016) 'A BPMN extension to enable the explicit modeling of task resources', in CEUR Workshop Proceedings, pp. 40–47.

BPMN-onto (2019) https://dkm.fbk.eu/bpmn-ontology, visited on 2019/03

Chungoora, N. et al. (2013) 'A model-driven ontology approach for manufacturing system interoperability and knowledge sharing', Computers in Industry. Elsevier, 64(4), pp. 392--401.

Dijkman, R. M., Dumas, M. and Ouyang, C. (2008) 'Semantics and analysis of business process models in BPMN', Information and Software Technology, 50(12), pp. 1281–1294.

Falbo, R. D. A. and Bertollo, G. (2009) 'A software process ontology as a common vocabulary about software processes', International Journal of Business Process Integration and Management, 4(4), pp. 239--250.

Fernandez, M., Gomez-Perez, A. and Juristo, N. (1997) 'METHONTOLOGY: From Ontological Art Towards Ontological Engineering', in Symposium on Ontological Engineering of American Association for Artificial Intelligence (AAAI), pp. 33--40.

Fraga, A. L., Vegetti, M. and Leone, H. P. (2018) 'Semantic interoperability among industrial product data standards using an ontology network', in 20th Int. Conference on Enterprise Information Systems (ICEIS). Madeira, Portugal, pp. 328–335.

Van Grondelle, J. C. and Gülpers, M. (2011) 'Specifying flexible business processes using pre and post conditions', in The Practice of Enterprise Modeling (PoEM). Oslo, Norway, pp. 38–51.

Grüninger, M. and Fox, M. S. (1995) 'The Role of Competency Questions in Enterprise Engineering', in Benchmarking—Theory and practice. Springer, Boston, MA, pp. 22–31.

Hepp, M. and Roman, D. (2007) 'An Ontology Framework for Semantic Business Process Management', Proceedings of Wirtschaftsinformatik. Universitätsverlag, 2007, pp. 1–18.

ISO (1998) 'ISO 10303-49 Industrial automation systems and integration -- Product data representation and exchange -- Part 49: Integrated generic resources: Process structure and properties'.

ISO (2005) '15531-32:2005 Industrial automation systems and integration -- Industrial manufacturing management data: Resources usage management -- Part 32: Conceptual model for resources usage management data'.

Karray, M. H., Chebel-Morello, B. and Zerhouni, N. (2012) 'A formal ontology for industrial maintenance', Applied ontology, 7(3), pp. 269–310.

Marcinkowski, B. and Kuciapski, M. (2012) 'A business process modeling notation extension for risk handling', in 11th Int. Conference on Computer Information Systems and Industrial Management (CISIM). Venice, Italy, pp. 374–381.

Natschläger, C. (2011) 'Towards a BPMN 2.0 ontology', in 3rd Int. Workshop on Business Process Modeling Notation. Lucerne, Switzerland, pp. 1–15.

OMG. (2011) Business Process Modeling Notation, v2.0 - Specification.

Rospocher, M., Ghidini, C. and Serafini, L. (2014) 'An ontology for the Business Process Modelling Notation', in 8th International Conference on Formal Ontology in Information Systems (FOIS). Rio de Janeiro, Brazil, pp. 133--146.

Roy, S., Dayan, G. S. and Devaraja Holla, V. (2018) 'Modeling industrial business processes for querying and retrieving using OWL+SWRL', in On the Move to Meaningful Internet Systems (OTM). Valletta, Malta, pp. 516–536.

Ruiz, F., Vizcaíno Barceló, A., Piattini, M., Garcia, F. (2004) 'An Ontology For The Management Of Software Maintenance Projects', International Journal of Software Engineering and Knowledge Engineering, 14(3), pp. 1–27.

Scheer, A.-W., Thomas, O. and Adam, O. (2005) 'Process Modeling Using Event-Driven Process Chains.', in Process-aware information systems, pp. 119–145.

Tartir, S. and Arpinar, I. B. (2007) 'Ontology evaluation and ranking using OntoQA', in 1st International Conference on Semantic Computing (ICSC). California, USA, pp. 185–192.

Uschold, M. et al. (1998) 'The Enterprise Ontology', The Knowledge Engineering Review, 13(1), pp. 31–89.

UO-onto (2019) http://purl.obolibrary.org/obo/uo.owl, visited in 2019/03

Vasilecas, O., Laureckas, E. and Rima, A. (2014) 'Analysis of Using Resources in Business Process Modeling and Simulation', Applied Computer Systems, 16(1), pp. 19–25.

Vogel-Heuser, B. and Hess, D. (2016) 'Guest Editorial Industry 4.0–Prerequisites and Visions', IEEE Transactions on Automation Science and Engineering, 13(2), pp. 411–413.

Wong, P. Y. H. and Gibbons, J. (2008) 'A Process Semantics for BPMN', in 10th International Conference on Formal Engineering Methods (ICFEM). Kitakyushu-City, Japan: Springer, Berlin, Heidelberg, pp. 355–374.