

# Semantic Annotation Framework to Manage Semantic Heterogeneity of Process Models

Yun Lin, Darijus Strasunskas, Sari Hakkarainen, John Krogstie and Arne Solvberg

Norwegian University of Science and Technology  
7491 Trondheim, Norway  
{yunl, dstrasun, sari, krogstie, asolvber}@idi.ntnu.no

**Abstract.** Effective discovery and sharing of process models within and/or across enterprises are important in process model management. A semantic annotation approach has been applied for specifying process semantic heterogeneity in the semantic process model discovery in our previous work. In this paper, the approach is further developed into a complete and systematic semantic annotation framework. Four perspectives are tackled in our framework: basic description of process models (profile annotation), process modeling languages (meta-model annotation), process models (model annotation) and the purpose of the process models (goal annotation). Ontologies, including modeling ontology, domain specific ontology and goal ontology, are used for annotation of process models to achieve semantic interoperability. A set of mapping strategies are defined to guide users to annotate process models.

## 1 Introduction

A considerable amount of business knowledge is put into process models and scattered within and across organizations. However, the possibility to efficiently retrieve and reuse this knowledge is limited. A research problem in process model management is how to cope with semantic heterogeneity of both process models and process modeling languages. The problem of semantic heterogeneity is even more critical in a situation of extensive cooperation and interoperation between distributed systems across different enterprises. The heterogeneity makes it difficult to manipulate the distributed process models in a centralized manner. Ontologies and semantic metadata provide a means to tackle this problem. Thus, we are using ontology-based semantic annotation to tackle the heterogeneous semantics of distributed process models.

In our previous work [10, 11], the semantic heterogeneity problems of process models have been analyzed and the ontology-based semantic annotation approach has been applied in example applications to test the feasibility of the approach. In this paper, the semantic annotation approach is further developed and refined in a semantic annotation framework, which contains profile annotation, meta-model annotation, model annotation and goal annotation.

In order to differentiate process models, we specify a set of metadata to annotate the significant characteristics of process models, terming this the profile. Ontology

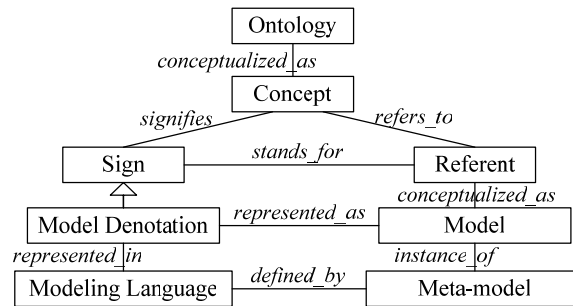
provides an alignment of the different terminology and conceptualization used in models and in modeling languages. Thus, we use ontologies to relate constructs across different modeling languages, as well as to align domain specific terminology used in models. In this way we are able to solve semantic heterogeneity in model management. Furthermore, in order to facilitate goal-driven process model reuse we annotate process models using a goal ontology. This allows discovery of process models used for achieving certain (business) goals.

The contribution of this paper is as follows. First, an extended and refined General Process Ontology (GPO) is presented that constitutes a semantic annotation framework. Second, a process semantic annotation model (PSAM) is developed and presented formally. This explicitly defines all necessary annotation elements. Third, mapping strategies and rules for process model annotation are described to provide a better guidance in model annotation.

The rest of the paper is organized as follows. In section 2, the theoretical basis on modeling and ontology of process models is discussed. Then, the semantic annotation framework and mapping strategies are described in details in section 3. In section 4, we define a process semantic annotation model and formalize it. In section 5 we compare the approach with the related work. Finally, we conclude this paper and outline our future work in section 6.

## 2 Theoretical Basis of Modeling and Ontology

In this section we will discuss the relationship between process models and ontologies in the context of semantic interoperability and semantic annotation.



**Fig. 1.** Relationship between ontology, model, meta-model and modeling language

We adapt the semantic triangle [14] to define the relationships between model, meta-model, modeling language and ontology (see Fig. 1). A model is a conceptualization of referents and it is represented as a set of model denotations in a certain modeling language. Model denotations are signs which signify concepts in the model. The model is an instance of a meta-model, and the meta-model defines a modeling language. Typically, a same concept can refer to different referents in different models. In order for the machine to understand the heterogeneous semantics of the models

(e.g. various signs of referents referring to the same concepts or synonymic signs of referents referring to different concepts), a common understanding of concepts has to be formulized in a machine-interpretable way. An ontology is created for this purpose. Here, concepts are conceptualized as an ontology for common understanding.

A meta-model is also a model – a model of the modeling language. Thus, a meta-model is the conceptualization of modeling referents referred by modeling concepts, and it can be concretized and represented as a specific modeling language. The modeling ontology is a kind of methodology ontology which contains a vocabulary of the modeling concepts (constructs). According to Leppanen’s OntoFrame [9], the meta-model can be adapted from the modeling ontology. Accordingly, the heterogeneous modeling languages can be aligned through annotating the meta-model by the modeling ontology.

### 3 Semantic Annotation Framework

In this section, we will describe our semantic annotation framework for process models in details. Four main annotation sets constitute the framework: namely, profile annotation, meta-model annotation, model annotation and goal annotation. They are discussed in more detail as follows.

#### 3.1 Profile Annotation

The basic and characteristic features of a process model are described by a set of metadata in the profile annotation. We categorize metadata elements for profile annotation according to the types of metadata – administrative, descriptive, preservation, technical and use [4].

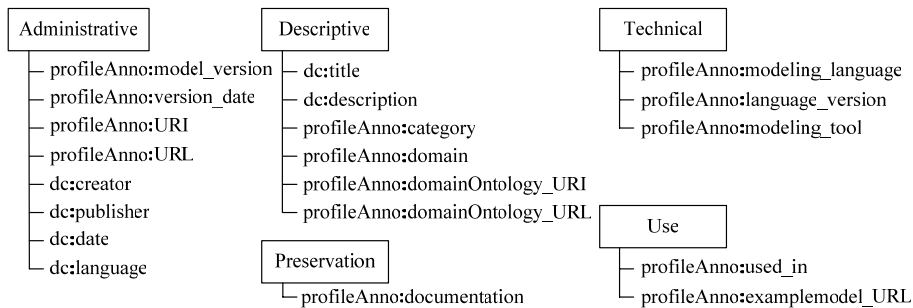


Fig. 2. Profile annotation metadata elements

We reuse some metadata elements from the Dublin Core metadata standard with prefix ‘dc’ and create also additional metadata with prefix ‘profileAnno’ to describe the profile of a process model. These metadata elements are classified in Fig. 2.

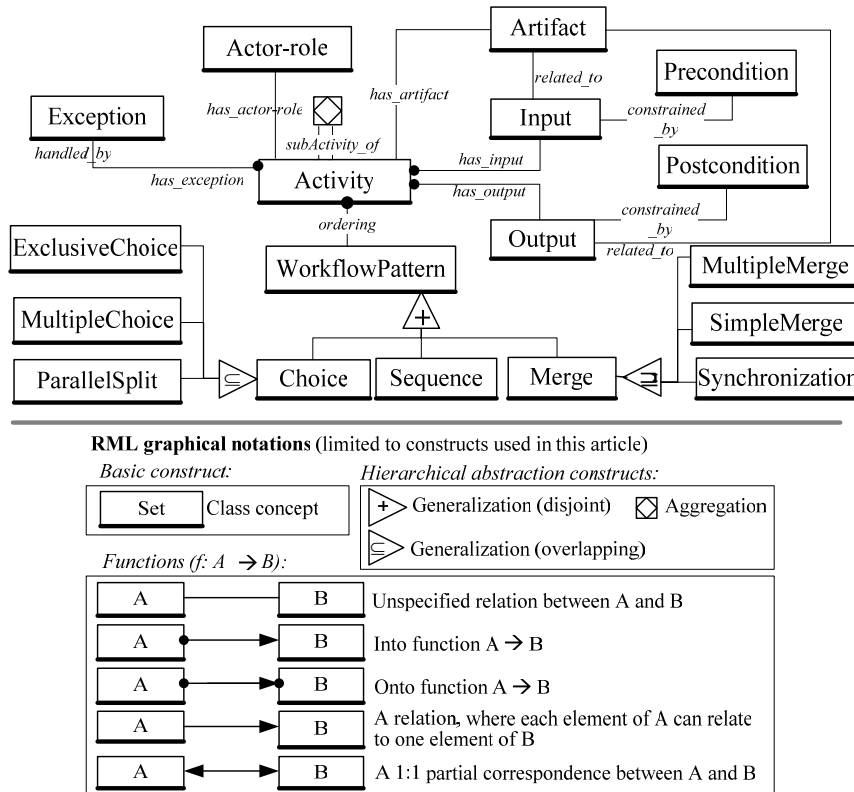
### 3.2 Meta-model Annotation

In the meta-model annotation, we use a process modeling ontology as metadata to annotate the semantics of constructs in a modeling language. Therefore, we first introduce our process modeling ontology, and then describe the way of annotating the semantics of meta-models of process modeling languages.

#### 3.2.1 Process modeling ontology – GPO

The process modeling ontology is used to align the heterogeneous meta-models of process models. The process modeling ontology should provide a common conceptualization of the concepts typically used in existing process meta-models or process modeling languages. In [10, 11], we built a General Process Ontology (GPO) according to the investigation of some process ontologies and process modeling languages including PSL, TOVE, PIF-CORE, CPR, EEML, BPMN and BPML. The visual GPO model is represented in RML (Referent Modeling Language) [17] in Fig. 3. RML is a conceptual modeling language, which can be used to visualize an ontology model.

In our GPO, we include the following concepts which are usually modeled as modeling constructs in most process modeling languages: *Activity*, *Artifact*, *Actor-role*, *Input*, *Output*, *Precondition*, *Postcondition*, *Exception* and *WorkflowPattern*. Compared with our previous work [10, 11], GPO is updated here by changing the relations between *Precondition*, *Postcondition* and *Activity* into the relations between *Precondition*, *Postcondition* and *Input*, *Output*. Such update is made because the pre- or post- conditions are directly related to inputs and outputs of activities. It is not necessary to make an indirect relationship between pre- or post-condition and input or output through the connection of activity. Another extension of GPO in this paper includes refinement of *WorkflowPattern* into several more specific patterns according to van der Aalst’s workflow patterns [19], such as *Choice* (*Exclusive Choice*, *MultipleChoice*, *ParallelSplit*), *Merge* (*SimpleMerge*, *MultipleMerge*, *Synchronization*) and *Sequence*, which are basic control workflow patterns supported by most process modeling languages with logical symbols like AND, OR and XOR. The aim of including workflow patterns in GPO is for the user to navigate the preceding, succeeding, synchronizing and exclusive activities, because those workflow patterns denote the semantics of process orders.



**Fig. 3.** Visualization of GPO

GPO is implemented as an ontology using Protégé according to the graphical GPO model in Fig. 3. The concepts in GPO will be mapped with meta-models of process modeling languages.

### 3.2.2 Mapping rules in meta-model annotation

GPO is a mediator for the semantics of process concepts and it should not be seen as a new process modeling language, but a means to annotate the process modeling constructs.

Meta-model annotation has to be done manually by experts who know the process modeling language to be annotated. The procedure of meta-model annotation is actually to set mapping rules between process modeling language constructs or meta-model elements and GPO. The mapping rules consist of both one-to-one and one-to-many correspondences between GPO concepts and modeling language constructs or meta-model elements. There may be more complicated cases: a correspondence between a GPO concept and a combination of some modeling language constructs or meta-model elements. To define the mapping rules for different cases, we categorize three types of modeling constructs – *AtomicConstruct*, *EnumeratedConstruct* and *ComposedConstruct*. Each single modeling language construct is an *AtomicConstruct*,

an *EnumeratedConstruct* is an enumeration set of several *AtomicConstructs*, and a *ComposedConstruct* is composed of several *AtomicConstructs*.

#### Mapping Rules:

- One-to-one mapping: a GPO concept (e.g. *GPO:Activity*) is referred by an *AtomicConstruct* (e.g. *EEML:Task*);
- One-to-many mapping: a GPO concept (e.g. *GPO:Artifact*) can be referred respectively by several modeling language constructs (e.g. *EEML:Information Object* and *EEML:Material Object*) which are enumerated in an *EnumeratedConstruct*;
- One-to-combination mapping: a GPO concept (e.g. *GPO:WorkflowPattern*) is referred by the combination of those modeling language constructs (e.g. *EEML:Flow* and *EEML:Decision Point*) in a *ComposedConstruct*.

A namespace **metaAnno** is used to encode meta-model annotation in these three mapping cases:

```
<metaAnno:AtomicConstruct rdf:ID="CONSTRUCT_ID">
  <metaAnno:refers_to
    rdf:resource="&GPO_ONTOLOGY#MODELING_ONTOLOGY_CONCEPT" />
  <metaAnno:modeling_language_construct
    rdf:resource="&MODELNG_LANGUAGE#LANGUAGE_CONSTRUCT" />
</metaAnno:AtomicConstruct>

<metaAnno:EnumeratedConstruct rdf:ID="CONSTRUCT_ID">
  <metaAnno:refers_to
    rdf:resource="&GPO_ONTOLOGY#MODELING_ONTOLOGY_CONCEPT" />
  <metaAnno:has>
    <metaAnno:AtomicConstruct rdf:resource="#CONSTRUCT_ID" />
    ...
  </metaAnno:has>
</metaAnno:EnumeratedConstruct>

<metaAnno:ComposedConstruct rdf:ID="CONSTRUCT_ID">
  <metaAnno:refers_to
    rdf:resource="&GPO_ONTOLOGY#MODELING_ONTOLOGY_CONCEPT" />
  <metaAnno:composed_of>
    <metaAnno:AtomicConstruct rdf:resource="#CONSTRUCT_ID" />
    ...
  </metaAnno:composed_of>
</metaAnno:ComposedConstruct>
```

Once the mapping rules are defined for a certain process modeling language, the process models in that process modeling language can be described by the GPO concepts, i.e. the GPO concepts are used as metadata to annotate the process semantics. We call the process models described by the GPO metadata as GPO-annotated process model. The GPO-annotated process model will be formulized in the *process semantic annotation model (PSAM)* in Section 4.

### 3.3 Model Annotation

When a process model is described by the semantic process annotation model, the process model looks similar to a Web service described by OWL-S. The semantics of the process can be interpreted by machine. However, the contents in the semantic process annotation model are only annotated with very abstract concepts like artifact, activity, actor-role, etc. The concrete domain information in the contents needs to be related with domain specific ontology in order to deal with the semantic heterogeneity of model contents. For example, two process models are both about travel booking domain. In one model, a concept called 'client' is annotated as an actor-role; while the concept named 'customer' is also annotated as actor-role in another model. If we map those two concepts to one concept in the travel booking domain ontology (in this case identifying them being synonyms), the machine will know how these two concepts are related, which helps to discover the related process model fragments. Therefore, we need to further annotate the model with domain specific ontologies. The model annotation facilitates the semantic discovery and navigation of process model fragments, which are parts of process models.

#### 3.3.1 Model annotation scope

The model annotation can be accomplished with the help of meta-model annotation, because the models related to domain information are usually artifacts, actor-roles, activities and exceptions in the semantic process annotation models. Artifacts and actor-roles are usually static concepts. Those concepts are defined in a local domain model which provides the context of those concepts. The model annotation is to map the concepts in the local domain model to the concepts defined in the domain specific ontology. If the concepts are not defined in a local domain model, the mapping has to be defined by the user manually. The activities and exceptions are usually related to the task ontology in certain domains. Currently, most domain specific ontologies consist of only static concepts including few concepts about activities or tasks. Task ontologies can be seen as reference processes. One example is SCOR<sup>1</sup> supported by SAP. Another example of task ontology is ontology of *Laboratory Procedures* from [8]: A *Laboratory Procedure* is a subclass of the class of *Health Care Activities* and a *Chemical* is a subclass of the class of *Substances*; a *Laboratory Procedure* analyzes a *Chemical* and a *Chemical* is analyzed by a *Laboratory Procedure*.

If the task ontology in a certain domain is available, the activities and exceptions can be annotated by concepts defined in the task ontology. If such task ontology can not be found, we can leave this part to the goal annotation, in which a set of predefined goals are linked to the processes or activities.

#### 3.3.2 Mapping strategies

Different mapping strategies can be used between concepts in the model and the domain specific ontology. It can be simple ones which are applied in meta-model annotation – by referring specific modeling constructs to corresponding domain concepts.

---

<sup>1</sup> SCOR(Supply-Chain Operations Reference-model), see [www.supply-chain.org](http://www.supply-chain.org)

It can also be more complicated through refined semantic relationships between concepts used in models and concepts defined in domain ontology.

**Simple reference.** If the simple mapping by reference is applied, it assumes that almost all concepts in the model have equal or approximately equal concepts in the ontology. The semantic relationship of mapping can be defined as one type – *refers\_to*. We have adopted such mapping strategy in the meta-model annotation to build the correspondences of concepts between modeling languages and the GPO. In the model annotation, users can choose this strategy provided the concepts in the models are very close to the concepts defined in the domain specific ontology. The strategy of simple reference is easy to apply to map the concepts and also make it easy for the machine to infer the mapping relationships without complicated algorithms.

**Refined semantic relationships.** Concepts used in process models are variously defined initially for different projects. Therefore, it might be difficult to find equally defined concepts in the domain specific ontology for process models. However, they are still within one domain, there must be some semantic relationships between concepts in models and concepts in ontology. In order to represent the semantic relationships precisely, we define some refined semantic relationships to link the concepts between models and ontologies for the model annotation.

As described in section 3.3.1 the contents related to artifacts, actor-roles, activities and exceptions are those to be annotated by domain specific ontology concepts. Semantic relationships and the corresponding annotation denotations generally used for the model annotation are listed in Table 1.

**Table 1.** Semantic relationships and corresponding annotation denotations

Semantic Relationship	Annotation Denotation
Synonym	<i>alternative_name</i> (terminology_level) <i>same_as</i> (concept level)
Polysemy	<i>different_from</i>
Hypernym	<i>kind_of</i>
Hyponym	<i>superConcept_of</i>
Meronym	<i>part_of</i> (Artifact) <i>member_of</i> (Actor-role) <i>phase_of</i> (Activity) <i>partialEffect_of</i> (Exception)
Holonym	<i>compositionConcept_of</i>

Both synonym and polysemy relationships are symmetrical. Hypernym and hyponym are inverse relationships. Concepts in the ontology are more general, while model concepts are relatively concrete for specific projects. Thus, *kind\_of* is more often used than *superConcept\_of* when annotating model concepts with ontology concepts. We provide more human sense expressions of meronymic relationship for artifacts, actor-roles, activities and exceptions respectively. For artifacts, we use *part\_of*, e.g. ‘Engine’ is a part of ‘Airplane’; for actor-roles, we use *member\_of*, e.g. ‘Airline’ is a

member of ‘Air Alliance’; for activities, we use *phase\_of*, e.g. ‘Flying’ is a phase of ‘Travelling’; for exceptions, we use *partialEffect\_of*, e.g. ‘Payment is cancelled’ is a partial effect of ‘Booking has failed’. The inverse relationship of meronym is holonymic relationship, which is seldom used in this framework because of the similar reason described for the hyponym relationship.

We focus on the type level process models in this research, the relationship between instance and class is not included in the framework. We assume that instances are already defined by type-level model constructs in each process model. The contents to be annotated are consequently on the type-level.

Since after the meta-model annotation the model can be described as a GPO-annotated process model, the model annotation can be done directly in the GPO-annotated process model instead of original models. Thereby, the GPO-annotated process model and the model annotation notations will be formalized in the *process semantic annotation model* in section 4.

### 3.4 Goal Annotation

Each process described in a process model is oriented towards achievement of certain goals. Goal analysis is initially separated from process modeling. Nonetheless, there is a relationship between goals and activities. As discussed in [12]: “Goals (desired states of the world) and activities (actions performed to achieve a particular state) are clearly different. However, analyzing activities and goals together makes clear the parallel between decomposing a goal into subgoals to be achieved and decomposing it into primitive activities to be performed”.

Some process modeling languages and tools support goal modeling as part of process modeling, such as EEML [7] implemented in the Metis tool<sup>2</sup>. Thus, a set of goals may be already modeled locally and linked to process models. Having an ontology of goals (global goals) will enable to build the relations between local goals and global goals.

In order to unify the semantics of the links between goals and process models and the relationships between local goal and global goal defined in goal ontology, we extend the annotation framework by additional links as follows. *Achieves\_lGoal* relationship between an activity and a local goal and *achieves\_gGoal* relationship between an activity and a global goal are used to denote that the activity can achieve the goal. Two types of relationships – *supports* and *contradicts* are defined for relationships between a local goal and a global goal. *Supports* means the achievement of the local goal will help to achieve the global goal; while *contradicts* discloses that achieving the local goal will prevent the realization of the global goal. The above relationships are formalized for the goal annotation in the *process semantic annotation model* in the following section.

---

<sup>2</sup> <http://www.troux.com/>

## 4 Process Semantic Annotation Model

As described earlier, a process model is represented by GPO concepts in the meta-model annotation. The content annotation and the goal annotation are applied on this GPO represented model. In this section, we formalize the GPO-annotated model together with the content annotation and the goal annotation as a process semantic annotation model.

**Definition 1.** Process Semantic Annotation Model (*PSAM*) contains concepts of GPO, domain specific ontology and goal ontology and is defined as follows.

$$PSAM = (AV, AR, AF, WP, I, O, \Theta^{pre}, \Theta^{pos}, E, PD, G^l, PG^g). \quad (1)$$

Where *AV* is a set of activities composing a process, *AR* is a set of actor-roles interacting with a process, *AF* is a set of artifacts participating in a process, *WP* is a set of workflow patterns, and each workflow pattern denotes an ordering of activities. *I* is a set of input parameters, *O* is a set of output parameters,  $\Theta^{pre}$  is pre-conditions when a process starts,  $\Theta^{pos}$  is post-conditions when a process ends, *E* is a set of possible exceptions occurring during a process. *PD* is a subset of domain ontology (*D*) concepts, i.e.  $PD \subseteq D$ , including static ontology concepts and task ontology concepts.  $G^l$  is a set of local goals and  $PG^g$  is a subset of goal ontology ( $G^g$ ), i.e.  $PG^g \subseteq G^g$ .

**Definition 2.** An activity can be considered as a simple process. Therefore an annotated activity is described as follows.

$$AV_i = (id, model\_fragment, name, alternative\_name, has\_Actor\_role, has\_Artifact, has\_Input, has\_Output, is\_in\_WorkflowPattern\_of, has\_Precondition, has\_Postcondition, has\_Exception, subActivity\_of, same\_as, different\_from, kind\_of, superConcept\_of, phase\_of, compositionConcept\_of, achieves\_lGoal, achieves\_gGoal). \quad (2)$$

Each element in *PSAM* has *id* and *name* to uniquely identify the element. *Model\_fragment* is the model fragment id in the original process model for keeping the link between the annotated model fragment and its annotation information. *Alternative\_name* provides synonym of the name from terminology level. Elements *has\_Actor\_role*, *has\_Artifact*, *has\_Input*, *has\_Output*, *is\_in\_WorkflowPattern\_of*, *has\_Precondition*, *has\_Postcondition*, *has\_Exception*, *subActivity\_of* denote the relationships between the activity and other related elements according to the GPO definition. The *ids* of the related elements are used in those relationships. We use *same\_as*, *different\_from*, *kind\_of*, *superConcept\_of*, *phase\_of*, *compositionConcept\_of* to annotate the activities with domain ontology, i.e. using semantic relationship mapping an activity with concepts defined in domain ontology. *Achieves\_lGoal* is to link the activity to the *id* of a local goal and *achieves\_gGoal* links the activity to a global goal defined in the goal ontology. Ontology concepts are denoted by URI (Uniform Resource Identifier) in the process semantic annotation model.

**Definition 3.** An actor-role is the person, agent or organization that interacts with an activity. The annotated actor-role is represented as follows.

$$AR_i = (id, model\_fragment, name, alternative\_name, same\_as, different\_from, kind\_of, superConcept\_of, member\_of, compositionConcept\_of). \quad (3)$$

**Definition 4.** An artifact is the thing consumed, used or produced in an activity.

$$AF_i = (id, model\_fragment, name, alternative\_name, same\_as, different\_from, kind\_of, superConcept\_of, part\_of, compositionConcept\_of). \quad (4)$$

**Definition 5.** A workflow pattern represents the type of the ordering of activities.

$$WP_i = (id, model\_fragment, name, alternative\_name). \quad (5a)$$

Refined workflow patterns are defined as follows.

$$Choice_i = (id, model\_fragment, name, alternative\_name, has\_inActivity, has\_outActivity, has\_logicConnector). \quad (5b)$$

Where the cardinality of *has\_inActivity* is 1. The *has\_logicConnector* element of *Exclusive Choice<sub>i</sub>*, *Multiple Choice<sub>i</sub>* and *ParallelSplit<sub>i</sub>* has value ‘XOR’, ‘OR’ or ‘AND’ respectively.

$$Merge_i = (id, model\_fragment, name, alternative\_name, has\_inActivity, has\_outActivity, has\_logicConnector). \quad (5c)$$

Where the cardinality of *has\_outActivity* is 1. The *has\_logicConnector* element of *Simple Merge<sub>i</sub>*, *Multiple Merge<sub>i</sub>* and *Synchronization<sub>i</sub>* has value ‘XOR’, ‘OR’ or ‘AND’ respectively.

$$Sequence_i = (id, model\_fragment, name, alternative\_name, has\_inActivity, has\_outActivity). \quad (5d)$$

Where the cardinalities of both *has\_inActivity* and *has\_outActivity* are 1.

**Definition 6.** Input and output are defined as parameters of an activity, which include value and data type. They are usually related to artifacts participating in the activity.

$$I_i = (id, model\_fragment, name, alternative\_name, data\_type, related\_artifact), \\ O_i = (id, model\_fragment, name, alternative\_name, data\_type, related\_artifact). \quad (6)$$

If a same artifact related with both input parameter and output parameter of an activity, the state of the artifact must be changed through this activity. We call it transformation.

**Definition 7.** Precondition and postcondition are presented by expressions to constrain input and output. The constraints are usually used as contract in services or process composition.

$$\Theta_i^{pre} = (id, model\_fragment, name, alternative\_name, related\_input), \\ \Theta_i^{pos} = (id, model\_fragment, name, alternative\_name, related\_output). \quad (7)$$

**Definition 8.** Exception happens in an activity and it can be handled by an activity.

$$E_i = (id, model\_fragment, name, alternative\_name, handler\_Activity, same\_as, different\_from, kind\_of, superConcept\_of, partialEffect\_of, compositionConcept\_of). \quad (8)$$

Exception will be annotated using predefined exception types in domain ontology. The activity handling the exception is pointed out by *handler\_activity*.

**Definition 9.** Local goals are sometimes defined together with local process models and linked to activities in the process model.

$$G_i^l = (id, model\_fragment, name, alternative\_name, supports, contradicts). \quad (9)$$

The relationships between a local goal and activities are defined in activity element. Here we only annotate the relationships between a local goal and a global goal with *supports* and *contradicts*.

Domain ontology ( $D$ ) and goal ontology ( $G^g$ ) are defined in current Web Ontology Languages, such as OWL. With the semantic process annotation model, the process semantics of different models can be caught and represented by the concepts of GPO, domain ontology and goal ontology which harmonize the semantic heterogeneity of process modeling languages, models and process goals. Based on the above formalizations, we define a markup process semantic annotation language with namespace **psam** by extending GPO ontological definitions in OWL using Protégé. Consequently, the extension includes adding a concept *LocalGoal* and properties for all concepts. An example of this markup annotation language of representing an annotated *Artifact* is below:

```
<psam:Artifact rdf:ID="ID">
  <psam:model_fragment rdf:resource="&MODEL_NAMESPACE#MODEL_ID" />
  <psam:name>NAME</psam:name>
  <psam:alternative_name>ALTERNATIVE_NAME</psam:alternative_name>
  <psam:same_as
    rdf:resource="&DOMAIN_ONTOLOGY#DOMAIN_ONTOLOGY_CONCEPT" />
  <psam:different_from
    rdf:resource="&DOMAIN_ONTOLOGY#DOMAIN_ONTOLOGY_CONCEPT" />
  <psam:kind_of
    rdf:resource="&DOMAIN_ONTOLOGY#DOMAIN_ONTOLOGY_CONCEPT" />
  <psam:superConcept_of
    rdf:resource="&DOMAIN_ONTOLOGY#DOMAIN_ONTOLOGY_CONCEPT" />
  <psam:part_of
    rdf:resource="&DOMAIN_ONTOLOGY#DOMAIN_ONTOLOGY_CONCEPT" />
  <psam:compositionConcept_of
    rdf:resource="&DOMAIN_ONTOLOGY#DOMAIN_ONTOLOGY_CONCEPT" />
</psam:Artifact>
```

## 5 Related work

Semantic interoperability is an active research area caused by the semantic heterogeneity in current information systems. Semantic interoperability is the ability to inte-

grate data sources developed using different vocabularies and different perspectives on data [16]. To achieve semantic interoperability, the semantics of data have to be machine-understandable.

The most popular approach to tackle the semantic interoperability problem is to apply domain ontology. The domain ontology approach uses a machine understandable definition of concepts and relationships between concepts so that there is a shared common understanding within a community [16]. Semantic annotation is a way of linking domain ontology and data to align the semantics defined heterogeneously into agreed machine-understandable semantics. It is initially applied in annotating unstructured Web content and digital documents e.g. Web pages, digital texts or images with formally defined semantic metadata. The semantic annotation helps to achieve more precise and efficient information retrieval on the Web or from Digital Libraries. Later on, the concept of annotation is extended to the structured Web Services to envision the Semantic Web Services. Semantically described services will enable better service discovery and allow easier interoperation and composition of Web Services [15]. The widely used Semantic Web Services Ontology Languages are DAML-S [1] and OWL-S [13] based on a W3C standard. Another emerging proposal for Semantic Web Services is from DERI<sup>3</sup> which comprises WSMO<sup>4</sup> (Web Services Modeling Ontology) and WSML (Web Services Modeling Language). Semantics of services content can be added to the services described either by syntactic Web Service standards or ontology-based description language. The common factor in most of these approaches is relating concepts in Web Services to domain specific ontologies [15], so called the semantic annotation approach.

The semantic annotation approach has been applied and tested on both unstructured and structured artifacts to achieve semantic interoperability. Seldom work is done on the semi-structured artifact, e.g. enterprise/business process models. In traditional information system development the process models were usually defined and used for a specific project within an enterprise. They were seldom reused in other projects or interoperated and integrated with other external process models. It is difficult to reuse models because of lacking of knowledge about the context of models, modeling methodologies, standards of modeling languages and also because of the semantic heterogeneity problem.

Some ongoing European projects like INTEROP [6] and ATHENA [2] aim to achieve the interoperability of heterogeneous systems and applications across networked enterprises. The interoperability of enterprise models is one of important issues dealt with in both projects. They share a common objective of Enterprise Modeling, i.e. providing a set of core modeling methodology elements or a shared language for supporting collaborative enterprise design and management. In the INTEROP project, a common enterprise modeling language – UEML 2.1 adapted from the UEML project [18] is under development. The UEML comprises languages and techniques that can be used for exchanging information between enterprise modeling tools [3]. Similarly, the POP\* methodology is proposed in ATHENA project including a set of common and basic modeling constructs to support model inter-

---

<sup>3</sup> <http://www.deri.org/>

<sup>4</sup> <http://www.wsmo.org/>

change. Although those two proposals are not directly associated with semantic annotation, both UEML and POP\* provides common semantic definitions of modeling constructs primarily for model exchange between tools and not for reusing existing models [5].

Our method deals with the heterogeneous semantic definitions of different process modeling languages by meta-model annotation through the General Process Ontology (GPO). Although GPO looks like UEML or POP\*, it is defined as an ontology only concerning the process dimension not a modeling language covering all perspectives in the enterprise domain. GPO is defined in OWL in order to make use of semantic Web technology to create a computer-interpretable semantic markup language for process modeling in the Web environment. As a whole, our semantic annotation framework intends to achieve the semantic discovery of process models but not to focus on the enterprise model translation as UEML or POP\* mainly do. Therefore, current UEML or POP\* methodology only deals with semantic heterogeneity on modeling language level. We have to address the semantic heterogeneity problem on both model and modeling language levels. Additionally the context of process models is also considered in our semantic annotation approach by employing the profile annotation and goal annotation.

## 6 Conclusions and Future Work

Based on our previous work, we have proposed a semantic annotation framework to manage the semantic heterogeneity of process models from the following perspectives: basic description of process models (*profile annotation*), process modeling languages (*meta-model annotation*), process models (*model annotation*) and purposes of the process models (*goal annotation*). Three ontologies are used for annotation purposes: *General Process Ontology* used for meta-model annotation, *domain ontology* for model annotation and *goal ontology* for process goal annotation. Furthermore, we have defined a set of mapping strategies for guiding users to annotate models. Nevertheless, the formal process semantic annotation model (PSAM) is the main contribution of this paper.

Further we are going to elaborate on the goal annotation part of the framework. A semantic annotation tool based on the framework is under development. We are looking at the possibility to make it as a Plug-in for Metis, a powerful tool for enterprise modeling and meta-modeling. The techniques of integrating or importing ontologies into modeling tools are a primary interest. The validity of the mapping in meta model annotation and model annotation will be checked using DL (Description Logic) based reasoning mechanisms. The framework is going to be further evaluated by implementing semantic process model discovery applications, and using this on selected case studies.

## References

1. Ankolekar, A., Burstein, M., Hobbs, J., Lassila, O., Martin, D., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Payne, T., Sycara, K.: DAML-S: Web service Description for the Semantic Web. In Proc. of the 1<sup>st</sup> Int'l. Semantic Web Conf., LNCS 2343, Springer-Verlag (2002) 348-363.
2. ATHENA Project (IST-2003-2004). <http://www.athena-ip.org> (2005) (Last accessed: 2006 02 15).
3. Bourrieres, J.P., Missikoff, M., Berre, A., Doumeingts, G., Piddington, C.: Deliverable D4.2 INTEROP 2<sup>nd</sup> Workplan. <http://interop-noe.org/deliv/d4.2/attach/D4.2%20V1.pdf> (2005) (Last accessed: 2006 02 15).
4. Gill, T., Gilliland-Swetland, A., Baca, M.: Introduction to Metadata: Pathways to Digital Information. Baca, M. (Ed.) Getty Information Institute, Los Angeles, USA (2000).
5. Grangel, R., Chalmeta, R.: A Methodological Approach for Enterprise Modeling of Small and Medium Virtual Enterprises based on UML: Application to a Tile Virtual Enterprise. In Proc. of Doctoral Symposium in the 1<sup>st</sup> Int'l Conf. on Interoperability of Enterprise Software and Applications, Geneva, Switzerland (2005).
6. INTEROP Project. <http://interop-noe.org> (2006) (Last accessed: 2006 02 15).
7. Krogstie, J., Jørgensen, D.H.: Interactive Models for Supporting Networked Organisations. In Proc. 16<sup>th</sup> Intl. Conf. on Advanced Information Systems Engineering, LNCS 3084, Springer-Verlag (2004) 550-562.
8. Kumar, A., Ciccicarese, P., Smith, B., Piazza, M.: Context-Based Task Ontologies for Clinical Guidelines. In Pisanelli, D.M. (Ed.) Ontologies in Medicine, Proc. of Workshop on Medical Ontologies. Amsterdam: IOS Press (2004) 81-94.
9. Leppanen, M.: An Ontological Framework and a Methodical Skeleton for Method Engineering: A Contextual Approach. PhD Thesis at University of Jyvaskyla, Finland (2005).
10. Lin, Y., Strasunskas, D.: Ontology-based Semantic Annotation of Process Templates for Reuse. In Castro, J. and Teniente, E. (Eds.): Proc. of the CAiSE'05 Workshops Vol.1 (EMMSAD Workshop). Porto, Portugal (2005) 593-604.
11. Lin, Y., Ding, H.: Ontology-based Semantic Web Annotation for Semantic Interoperability of Process Models. In Proc. of the Int'l Conf. on Computational Intelligence for Modeling, Control and Automation, Vienna, Austria (2005).
12. Malone, T.W., Crowston, K., Herman, G.A.: Organizing Business Knowledge: The MIT Process Handbook. The MIT Press (2003).
13. Martin, D., Burstein, M., Hobbs, J., Lassila, O., McDermott, D., McIlraith, S., Narayanan, S., Paolucci, M., Parsia, B., Payne, T., Sirin, E., Srinivasan, N., Sycara, K.: OWL-S: Semantic Markup for Web Services. <http://www.w3.org/Submission/OWL-S/> (Last accessed: 2006 02 15).
14. Ogden, C., Richards, I.: The Meaning of Meaning. London: Kegan Paul (1923).
15. Patil, A., Oundhakar, S., Sheth, A., Verma, K.: METEOR-S Web Service Annotation Framework. In Proc. of the 13<sup>th</sup> Int'l. World Wide Web Conf., ACM Press (2004) 553-562.
16. Ram, S., Park, J.: Semantic Conflict Resolution Ontology (SCROL): An Ontology for Detecting and Resolving Data and Schema-Level Semantic Conflicts. IEEE Transactions on Knowledge and Data Engineering 16(2), (2004) 189-202.
17. Solvberg, A.: Data and What They Refer To. In Chen, P., Akoka, J., Kangassalo, H., Thalheim, B. (Eds.): Conceptual Modeling: Current Issues and Future Trends. LNCS 1565. Springer-Verlag (1999) 211-226.
18. UEML Project (IST-2001-34229). <http://www.ueml.org> (2005) (Last accessed: 2006 02 15).
19. van der Aalst, W.M.P., Barros, A.P., ter Hofstede, A.H.M., Kiepuszewski, B.: Advanced Workflow Patterns. In Proc. of the 7<sup>th</sup> Int'l. Conf. on Cooperative Information Systems, LNCS 1901, Springer-Verlag (2000) 18-29.