Ontology-based Semantic Annotation of Process Templates for Reuse

Yun Lin, Darijus Strasunskas

Dept. Of Computer and Information Science
Norwegian Univ. of Science and Technology
7491 Trondheim, Norway
{Yun.Lin,Darijus.Strasunskas}@idi.ntnu.no

Abstract. Process templates are stored as valuable resources and then are retrieved and reused in other projects. In order to find a desired template, the semantics of various process templates should be machine-readable and interoperable. However, the heterogeneity of both model representations and modeling languages makes it difficult to reuse the templates. Here we adopt one of the emerging semantic web techniques – the semantic annotation of process templates in order to enhance the interoperability for better reuse of process templates. Our semantic annotation consists of three basic parts: model profile, model content and meta model annotation. A general process ontology and domain ontologies are referenced as the annotation information. Given process templates annotated by ontology, they are abstracted from language-specific details but to the level of necessary details for process templates to be reused.

Keywords: process modeling, process template, reuse, semantic annotation, process ontology

1 Introduction

Process templates provide reusable process model structures and they can be instantiated and tailored to specific requirements. They serve as knowledge and resources of legacy system for further reuse. Model template reuse improves the quality of process models by reflecting previous knowledge and experience preserved in them. To be useful and usable, a desired process template should be easily located and adapted in a new project. This requirement is easy to satisfy within the boundaries of one enterprise using the same modeling environment. However, sharing and exchanging knowledge and resources across enterprises and different domains become more and more intensive. Process templates from different enterprises or projects are presumed to provide reusable process modeling experience to any new project. Therefore, we focus on enhancing storage and retrieval of process templates in this paper. One critical issue in this application is the interoperability [15] of heterogeneous models due to various modeling languages and representations.

We distinguish the interoperability problems from a model and a meta model level. Two typical interoperability problems exist on both levels: 1) Terms are used differently for the same concept, on the model level, e.g., ‘Client’ vs. ‘Customer’,

'purchase' vs. 'buy'; on the meta model level, e.g., one of model constructs is called 'agent' in ActionWorkflow [11], but this concept is called 'actor' in CPR (Core Plan Representation) [13]. 2) Conceptualization is mismatched, on the model level, e.g., Class (city) vs. Property (city), Action (finish) vs. State (finish); on the meta model level, e.g., in PSL (Process Specific Language) [14] 'activity' is defined as the atomic concept but it is not an atomic concept in WooRKS [1].

A common understanding of model representations is needed to enhance interoperability when searching process templates. All process templates are annotated based on domain information as well as shared common process ontology. The underlying assumption is that process modeling languages have sufficient similarities so they can sensibly be mapped to more abstract common constructs.

We develop a common semantic annotation structure for the process templates in three parts: model profile annotation, model content annotation and meta model annotation. The three parts stand for three perspectives of models – the model as a whole product, model fragment and modeling language.

In section 2 we introduce a semantic annotation structure for the process template. For supporting the semantic annotation, a general process ontology and a set of common process template modeling morphemes are proposed in section 3. In section 4, we illustrate how to apply the semantic annotation structure to a process template together with the proposed process ontology and modeling language. Finally, we draw the conclusions.

2 Semantic Annotation of Process Template

A common understanding of the heterogeneous semantics should be agreed to target the interoperability of process templates. Semantic annotation is one of techniques by adding metadata and using a set of ontology to describe the semantics of information. We build a common semantic annotation structure and apply process ontology and domain ontology to enhance the interoperability of process templates retrieval.

Three aspects annotate: model profile (for context semantics interoperability), model content (for model fragment semantics interoperability) and meta model (for modeling language semantics interoperability).

2.1 Model Profile Annotation

The metadata describing a model template as a whole is called profile[1], such as the name of the model template, the author/creator of the template, date of the template' creation, problem domain of the template and etc. The profile structure can refer to the description structure of patterns [8] including name, problem, context, solution, example, consequences [16], forces, related patterns, or known uses [3].

[1] Note that the definition of profile here is not same as the UML profile. A UML profile identifies a subset of the UML meta model. (Refer to the UML profile at http://lists.ebxml.org/archives/ebxml-bp/200002/msg00008.html).

2.2 Model Content Annotation

We divide the content of a process template into two parts: the process and the domain. The process part is the workflow description. Any objects participating in the process are defined and represented in a domain model separated from the process part. The domain model is an object-relation model. Locally the semantics of the models are explicitly represented using the modeling languages. The shared semantics have to be annotated with the reference ontology. Process concepts and process patterns defined in the reference process ontology annotate the model fragments in the process part.

We assume all the process templates can be exported or built in XML/RDF files. The markup element is annotated by attaching a concept with a prefix of the reference ontology as shown in Figure 1.

```
<semAnn:concept rdf:resource="REFERENCE_ONTO#CONCEPT">
    <ELEMENT id="…">
       …
     </ELEMENT>
</semAnn:concept>
```

Fig. 1. Cutout of Model Semantic Annotation

If **ELEMENT** is a process modeling construct in the process part of the process template, **REFERENCE_ONTO#CONCEPT** is a concept of the reference process ontology. If **ELEMENT** is an object-relation modeling construct in the domain part, **REFERENCE_ONTO#CONCEPT** is a concept of the reference domain ontology in the domain part.

2.3 Meta Model Annotation

Meta model annotation provides semantic harmony of modeling languages. Different model constructs are semantically abstracted and mapped to a set of model morphemes for the process template. For example, ‘actor’ in one process modeling language is semantically equal to ‘agent’ in another process modeling language. However, in the set of model morphemes for the process template, the concept ‘actor’ is defined as a common modeling construct. ‘Agent’ or ‘actor’ defined in the specific modeling languages is annotated using the agreed construct ‘actor’.

3 Ontological basis for Process Templates

Our approach is ontology-based annotation of the process templates. As common reference point, the process template ontology should be on a type-level. The type-level is grounded at the instance-level which provides the semantics [2]. At the type-level, we introduce General Process Ontology (GPO) to describe process templates.

Originally, a process template is abstracted from a specific process model in some project and it is made in a specific process modeling language. In order to represent templates and store them with structural constructs, a neutral process template modeling language is derived following our GPO. Comparing with certain process modeling languages, the Process Template Modeling Language (PTML) tries to cover the common and core constructs used in most process modeling languages and also tries to simplify the structures of models with the several modeling constructs. Although it is thought to be a new process modeling language, it is not created for that purpose. PTML is a result of further investigation for our assumption that process languages have sufficient similarity as to be sensibly mapped to more generic common constructs. It is used to represent the retrieval results of process templates independent of specific modeling tools. The details of the usage of the GPO and PTML will be discussed in the section 4.

3.1 General Process Ontology (GPO)

We build a General Process Ontology (GPO) based on BWW (Bunge-Wand-Weber) ontology [19]. BWW ontological constructs provide semantic basis of meta models of conceptual models. BWW ontology can be used as the upper level ontology because BWW ontology model is initially built as a set of core constructs that underlie the computer science and information systems fields [18], especially the core concepts used in conceptual modeling, namely *Thing*, *Property, State, Law, Event, Process, Transformation* and *System*. Although it is not initially process-oriented, some concepts are related with dynamic parts of information systems, such as *Event*, *State*, *Process* and *Transformation* which we call the process concepts. They are major process concepts which can be represented by most process modeling languages (PMLs) such as PSL [14], TOVE [7], PIF-CORE [10], CPR [13], APM [6], EEML [9], BPMN [5] and BPML [5]. Through investigation of those PMLs, we adapt BWW ontology into a GPO. The general process ontology is represented by RML (Referent Modeling Language) [17] in Figure 2.

We found ***Activity*** is often used in most process ontology or modeling languages. In set of process modeling languages, *Activity* is defined as a composition of events or operations. Comparing with *Activity*, concept *Event* is a detailed analysis concept. According to Bunge [4], processes may be either chains or trees of events. From this perspective, *Activity* is a synonym of *Process*. However, we also found *Process* is seldom a construct or just a package construct in most process modeling languages because it is obvious that a process model describes processes. Since in a business or enterprise process template we do not need to concentrate on the detailed sequence of *Event*, we therefore use concept *Activity* in our general process ontology. An activity may be an atomic activity or a composed activity represented by the aggregation relation between activities, i.e. one activity can be a part of another activity. Moreover, one activity may be a kind of another activity, e.g. ‘swallow’ is a kind of ‘eat’. The relation of the ‘kind of’ is represented by semantic *is_kind_of* in the GPO.

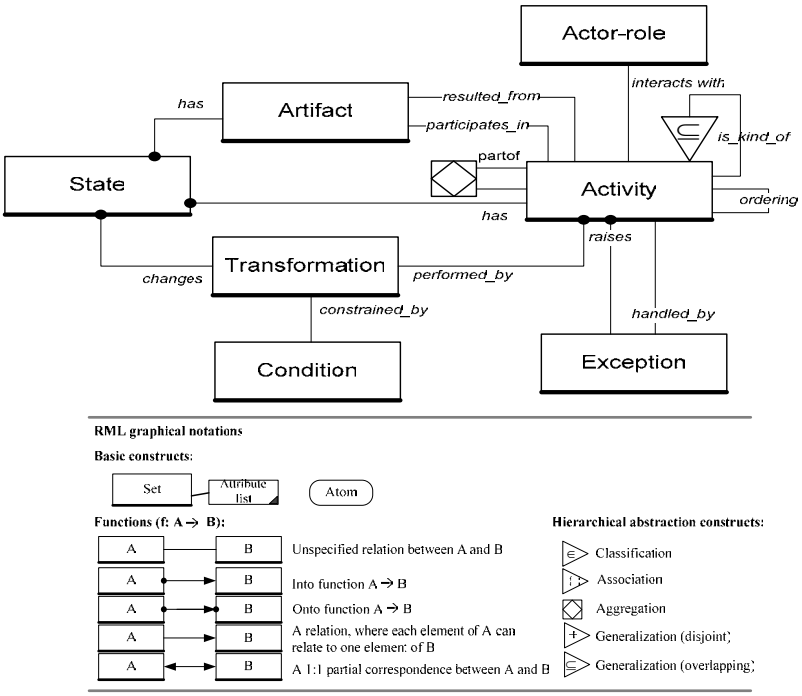

Fig. 2. The General Process Ontology (GPO)

Artifact represents something involved in an activity such as product, information, tool and software. Artifacts are not specified in more details. We only differentiate the direction of the relations between *Artifact* and *Activity*, i.e., 'participate_in' and 'resulted_from'. In a model template, only inputs and outputs are specified. ***Actor-role*** is the one who interacts with the activity. Although actor and role are two different concepts, we combine those into one. *Actor-role* does not represent an instance of actor or role in process template but on a class level.

State is the core concept in state modeling languages, e.g. statechart. State models are closely related to process models although the two kinds of models are not described in a same modeling language. In [19], *state of a thing* is described as "the vector of values for all property functions of a thing" in the BWW ontology. We simplify it by saying that *Artifact* has *State*. In a process model, *State* is usually used together with an *Activity*, e.g. 'start', 'finished', 'suspended'. If we consider an activity is also a thing, we can extend the BWW ontology that an *Activity* may have *State*. Another concept ***Transformation*** defined in the BWW ontology is also seldom used as a construct in a process modeling language but it stands for a phenomenon in all processes. A *Transformation* can change *State* and a *Transformation* is performed by an *Activity*.

Condition represents some context of a process which constrains a *Transformation*. ***Exception*** provides additional information about the failure of the process or any exceptional cases in a process.

3.2 PTML Meta Model

A meta model is a way to interpret a modeling language. The meta model of the Process Template Modeling Language (PTML) is displayed in Figure 3. The elements in the meta model of the PTML are process template model morphemes.

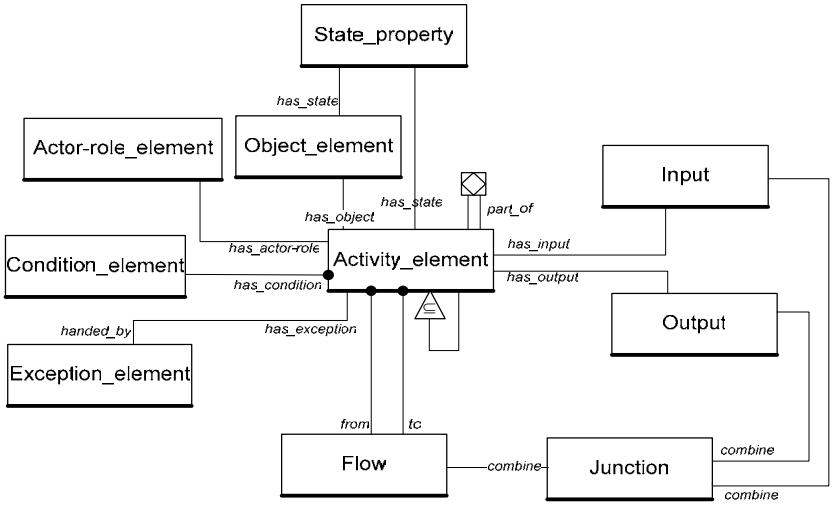

Fig. 3. PTML Meta Model

PTML is derived from the GPO and also references to some process modeling languages, such as PSL, APM, EEML, BPMN, BPML etc. Some model morphemes are directly mapped from the concepts in GPO and we just distinguish them by putting a postfix '_element' in PTML. Such model morphemes are `Activity_element`, `Actor-role_element`, `Artifact_element`, `Condition_element` and `Exception_element`. In the various process modeling languages, the similar constructs to `Activity_element` are 'Action', 'Task', 'Activity' etc. The possible `Actor-role_element` in other process modeling languages may be 'Agent', 'Role', 'Actor', 'Object' or 'User'. 'Object', 'Artifact', 'Resource', 'Product' or 'Tool' defined in some process modeling languages can be mapped with `Artifact_element` in PTML. Since *State* is always described as a state of an artifact or a state of an activity, it is difficult to represent *State* as an independent construct. *State* is therefore a property of *Artifact* and *Activity*. `Condition_element` is usually presented as 'PreConditon', 'Constraint', 'Rule' and etc. `Exception_element` can be refined as 'Error Handler', 'Fault Handler' in certain process modeling languages.

There are also some model morphemes such as `Input`, `Output`, `Flow` and `Junction` which are commonly used in many process modeling languages. Those model morphemes are not represented in GPO, since they are not the concepts that are often used in describing a process in our daily life. However, they are necessary

elements to build a model. `Input` and `Output` are ports of the `Activity_element` and they provide the interface for parameters in an activity. `Flow` is used to link activities and shows the sequence of the activities. `Junction` is a logic connection for joining or splitting flows, inputs or outputs. Combining those model morphemes can represent semantics of GPO. For example, the `Activity_element` and `Flow` can present the *ordering* of the *Activities*. If there are two activities are linked by a flow and the properties of the flow are `from` and `to`, the model is represented by *Flow_1*<from> *Activity_1* and *Flow_1*<to> *Activity_2*. Such model discloses the semantics that *Activity_1* is preceding of *Activity_2*.

Transformation does not have corresponding model morphemes in PTML. Since *Transformation* is performed by *Activity* and changes of the *States*, the semantics can be interpreted by `Activity_element` together with `State_property` and other model morphemes.

It is not possible to build one to one mapping between PTML and a specific process modeling constructs. However, PTML can represent the core of the process by template modeling in order to store and present process templates in a simple and comprehensible way for normal users.

4 Applying the Semantic Annotation to a Process Template

In this section, we will demonstrate how to annotate the process template with GPO and describe the process template in PTML. An application of the semantic retrieval of process templates is also briefly presented here.

4.1 Semantic Annotation for a Process Template

Figure 4 illustrates the architecture of the semantic annotation for a process template. The start is a local process template. As we state in previous sections that a model template is assumed to have two parts: the process and the domain part. The process part defines and describes the semantics of a process which is represented by a specific process modeling language in a project. The process part can be presented by the general process ontology. Although objects and actors are also involved in the process part, they are defined in the domain model and linked to the process part. The domain model defines the classes, attributes, and relationships of objects and actors from the static perspective. They are referred in the process part through 'ID' defined in the domain model. The domain model can be thought as a local domain ontology. The structures involving 'Local process template', 'Meta model of a specific process modeling language' and 'Local domain model' are shown on the left side of the figure.

A simple example illustrates the structures in Figure 5. There is a process template of a buying process. The specific process modeling language is EEML. In this EEML process template, 'purchase' is a task and 'client' is a personrole. The task 'purchase' is defined in the process part. Although 'client' is used as a personrole in the process part but it is defined as a Class in local domain model (ldm). Hence the ID of 'client' defined in the local domain model is used for personrole by URI

(Uniform Resource Identifier). The process part is in XML and the domain model is in RDFS. The meta model of EEML is described in a XMLS (XML Schema) file.

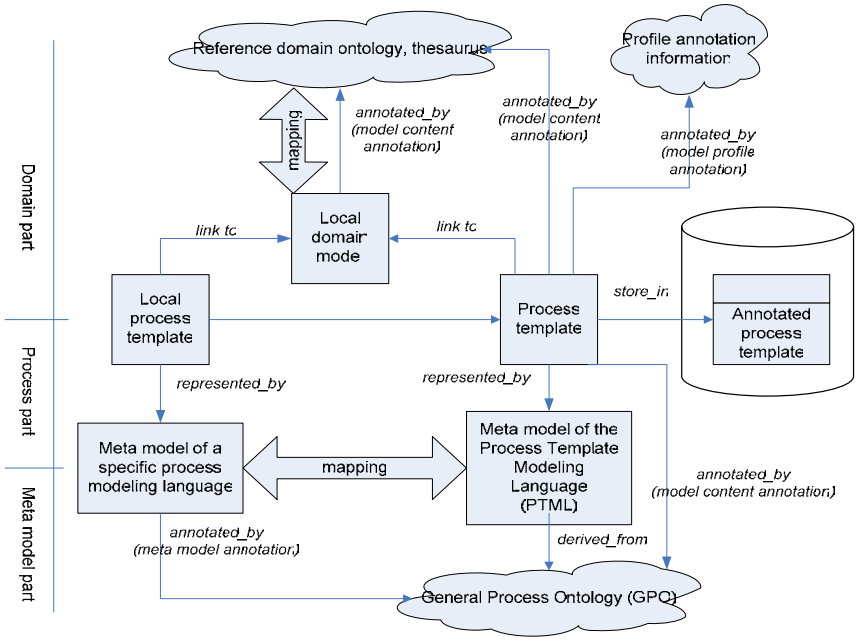

Fig. 4. Architecture of Semantic Annotation for a Process Template

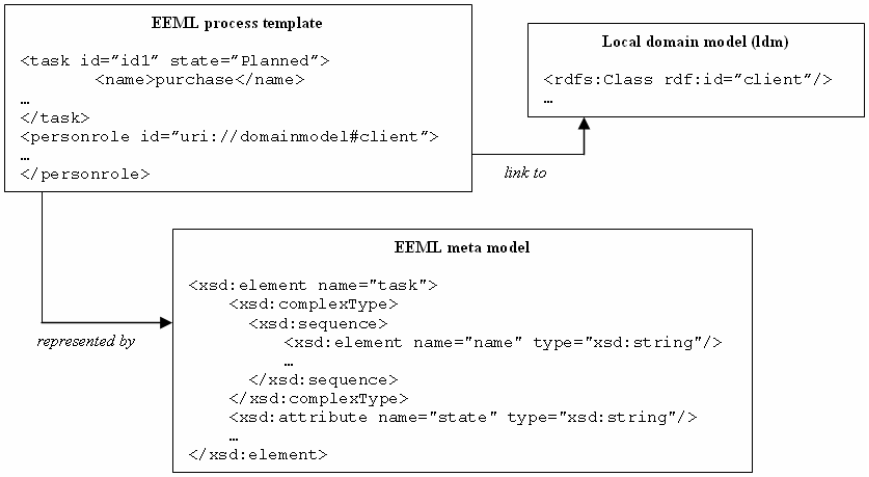

Fig. 5. Structures of EEML Process Template

Meta model annotation

In order to share the local process template, it has to be described in a common language with enough semantic information. A mapping between a specific process modeling language and PTML supports describing templates in a common modeling language for showing the templates to users. Such mapping can be done guided by the GPO. The original specific process modeling language constructs are annotated by concepts of GPO. According to the semantic annotation of modeling construct, the corresponding model morphemes of PTML will be used to displace the original modeling construct. The semantic annotation here is the ***meta model annotation***.

Since PTML normally contains less constructs than any specific process modeling language, the mapping from a concrete set to a general set is relatively easier than the reverse mapping. The semantics loss is obviously hard to avoid in the mapping. Fortunately a process template is not so complicated or concrete as a model instance. A general description of the model template represented by a specific construct can be described using a relatively general concept or construct without losing too much of original semantics.

Mapping of two meta models is a kind of ontology mapping if we consider a meta model being the ontology of a modeling language. There are already many ontology mapping techniques. Here we use SKOS[2] Core to describe the ontology mapping. SKOS Core is an RDF schema for representing thesauri and similar types of knowledge organization system (KOS). More information about SKOS Core can be found in [12].

In the EEML buying process example, element task is annotated with concept *Activity* in GPO in SKOS representation (Figure 6). According to the semantic annotation and comparing meta models of EEML and PTML, construct task in EEML can be replaced by 'Activity_element' in PTML.

Meta model annotation

```
<skos:Concept rdf:about="uri://eeml#task">
    <skos:preLabel>task</skos:preLabel>
    <skosmap:broader rdf:resource="uri://processonto#Activity"/>
</skos:Concept>
```

Fig. 6. Meta Model Annotation

Model content annotation

In this part, we distinguish two steps of content annotation for both – domain model content and process part. Since a process template does not contain any instance and all the concepts are on a type-level, the domain model is a local ontology or thesaurus for the project. The concepts used in the local ontology can be annotated with concepts used in a reference domain ontology or thesaurus by mapping two ontologies. 'Client' defined in the local domain model can be annotated with an agreed reference domain ontology. We assume that there is a concept 'Customer', and term 'client' is defined as its synonym in the reference domain ontology. We can use expression in Figure 1 to annotate the local domain model. In addition, we use SKOS

[2] Simple Knowledge Organization Systems http://www.w3.org/2004/02/skos/

to annotate ‘client’ appearing in the local domain model for more refined semantics mapping. Figure 7 shows such content semantic annotation using SKOS.

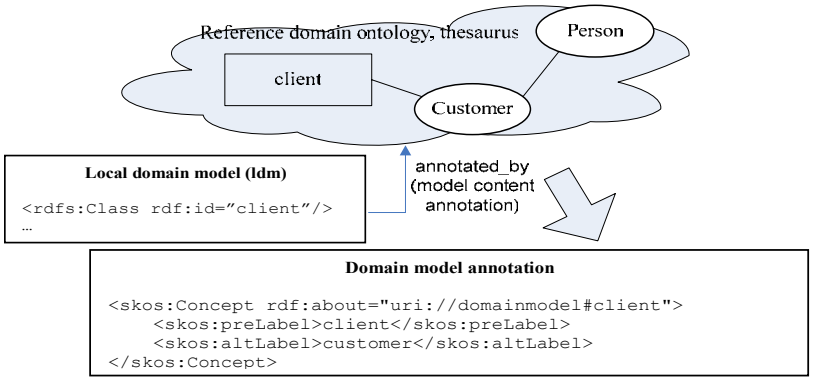

Fig. 7. ‘Client’ in Local Domain Model is Annotated with Reference Domain Ontology

The process part is partially annotated by meta model annotation as the model constructs are annotated by GPO. However, the model fragments combined with several model constructs need further semantic annotations using GPO. In addition to GPO, the specific terms used in the template can also be annotated using thesaurus. Generalization and aggregation relations of activity can also be annotated referring the reference domain ontology or thesaurus. Through the meta model annotation, we already know that task in the process part is replaced by ‘activity_element’ in process template. A model fragment containing ‘activity_element’ describes an *Activity* (in GPO) and its semantics can also be annotated. Figure 8 represents the model content annotation for the process part. For the model fragments, we do not adopt the SKOS method but the expression in Figure 1 because such an annotation is an explanation of a model fragment not a concept mapping.

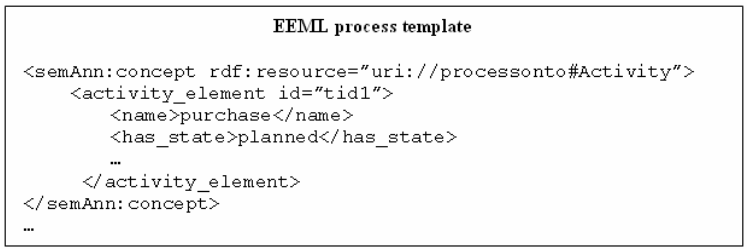

Fig. 8. Model Content Annotation for the Process Part

Model profile annotation

Before the process template is stored in a repository, some basic descriptions of the template should be annotated in the model profile as metadata.

4.2 Semantic Retrieval of Process Template

The purpose of semantic annotation for the process template is to enhance their interoperability in semantic retrieval of process templates. The semantic retrieval includes not only finding a process template as a whole based on the profile information, but also retrieving of process templates which are most relevant according to the model content annotation and the meta model annotation.

Users can choose predefined problem domain or problem category as a query condition, and users may also just enter keywords for searching. Before querying, the keywords will be matched firstly with concepts in GPO ontology and reference domain ontology. The search is then executed using extended set of keywords and reasoning based on GPO ontology as well as reference domain ontology.

5. Conclusions

Interoperability is a very vital issue and a difficult problem in meaning understanding and sharing in many enterprise applications, such as heterogeneous enterprise application integration, e-business, web services, knowledge sharing and information exchanging. In this paper, we address an interoperability problem in semantic retrieval of process templates for reuse purpose. Since the process templates are reusable knowledge and valuable resources of process modeling, they are required to be comprehensible and adaptable to other enterprise modeling users. We discern two main levels of interoperability of model templates, namely model level and meta model level. To enhance reuse of process templates, we propose a semantic annotation method to annotate templates, process model fragments, and modeling languages.

The main contribution of this approach is the General Process Ontology which is used to annotate common and general process concepts in the Universe of Discourse. GPO is used as a referent process ontology in process modeling. A set of process model morphemes in a neutral Process Template Modeling Language is generated from GPO and is used to represent the process template. GPO and PTML are not designed for very specific or detailed models but a model template, so the concepts and the constructs are more general and simple than some process modeling languages. It can be regarded as an approach allowing generalization of process models to the level of necessary details for templates to be reused. Moreover, the paper presented the semantic annotation structure and discussed the architecture of our approach. It disclosed the technical possibility of the approach. With the semantic annotation of process templates, process model designers can retrieve their desired templates and reuse them in their specific projects.

Acknowledgements. The authors thank John Krogstie for his helpful comments on this paper.

References

1. Ader, M., Lu, G., Pons, P., Monguio, J., Lopez, L., De Micelis, G., Grasso, M. A. and Vlondakis, G.: WooRKS, an Object Oriented Workflow System for Offices, ITHACA technical report. Available at ftp://cui.unige.ch/OO-articles/ITHACA/WooRKS, (1994)
2. Aitken, S., Curtis, J.: Design of a Process Ontology: Vocabulary, Semantics, and Usage. In Proceedings of the 13th Intl. Conf. on Knowledge Engineering and Knowledge Management, Springer Verlag, (2002)
3. Appletion, B.: Patterns and Software: Essential Concept and Terminology (2000). Available at: http://www.cmcrossroads.com/bradapp/docs/patterns-intro.html
4. Bunge, M.: Treaties on Basic Philosophy, Vol.3, Ontology I: The Furniture of World. Boston, MA: Reidel (1977)
5. Business Process Management Initiative. Available at: http://www.bpmi.org/ (2000)
6. Carlsen, S.: Conceptual modeling and composition of flexible workflow models. PhD thesis, NTNU, Trondheim, Norway (1997)
7. Fox, M.S., The TOVE Project: A Common-sense Model of the Enterprise, Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, Belli, F. and Radermacher, F.J. (Eds.), LNAI, Springer Verlag, (1992)
8. Gamma, E., Helm, R., Johnson, R., Vlissides, J., Design Patterns: Elements of Reusable Object-Oriented Software. Addison Wesley Professional. ISBN: 0201633612, (1994)
9. Krogstie, J. and Jørgensen, D. H., Interactive Models for Supporting Networked Organisations. In Proceedings of 16th Intl. Conf. on Advanced Information Systems Engineering (CAiSE'04), Springer Verlag, LNCS 3084, Riga, Latvia, (2004)
10. Lee, J., Grunninger, M., Jin, Y., Malone, T., Tate, A., Yost, G., and other members of the PIF Working Group The PIF Process Interchange Format and Framework Version 1.1, May (1996)
11. Medina-Mora, R., Winograd, T., Flores, R., and Flores, F., The Action Workflow Approach to Workflow Management Technology, In Proceedings of the Conf. on Computer Supported Cooperative Work (CSCW'92), Canada, ACM, (1992)
12. Miles, A.J., Rogers, N., Beckett, D. Skos-core 1.0 guide. Available at: http://www.w3.org/2001/sw/Europe/reports/thes/1.0/guide/, (2004)
13. Pease, A., Carrico, T.M.: Object Model Working Group (OMEG) Core Plan Representation – Request for Comment, version 2, DARPA, (1997)
14. Schlenoff, C., Gruninger, M., Tissot, F., Valois, J., Lubell, J., Lee, J., The Process Specification Language (PSL) Overview and Version 1.0 Specification. NISTIR 6459, National Institute of Standards and Technology, Gaithersburg, MD., (2000)
15. Sheth, A.: Changing Focus on Interoperability in Information Systems: from System, Syntax, Structure to Semantics. Interoperating Geographic Information Systems. Goodchild, M.F., Egenhofer, M. J., Fegeas, R. and Koffman, C.A. (eds), Kluwer (1998)
16. Stevens, P., Pooley, R.: Systems Reengineering Patterns. In Proceedings of ACM-SIGSOFT, 6th Intl. Symposium on the Foundations of Software Engineering, pp.17-23, ISBN 1-58113-108-9, (1998)
17. Sølvberg, 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)
18. Wand, Y., Weber, R., An Ontology Model for an Information System. IEEE Transactions on Software Engineering, 16 (11), (1990)
19. Wand, Y., Weber, R.: On the Deep Structure of Information Systems. Information System Journal, 5:203-223, (1995)