International Journal of Business, Human and Social Sciences
ISSN: 2517-9411
Vol:1, No:12, 2007

Using Ontology Search in the Design of Class Diagram from Business Process Model

Wararat Rungworawut, and Twittie Senivongse

***Abstract*—**Business process model describes process flow of a business and can be seen as the requirement for developing a software application. This paper discusses a BPM2CD guideline which complements the Model Driven Architecture concept by suggesting how to create a platform-independent software model in the form of a UML class diagram from a business process model. An important step is the identification of UML classes from the business process model. A technique for object-oriented analysis called domain analysis is borrowed and key concepts in the business process model will be discovered and proposed as candidate classes for the class diagram. The paper enhances this step by using ontology search to help identify important classes for the business domain. As ontology is a source of knowledge for a particular domain which itself can link to ontologies of related domains, the search can give a refined set of candidate classes for the resulting class diagram.

***Keywords*—**Business Process Model, Model Driven Architecture, Ontology, UML Class Diagram.

I. Introduction

MODEL Driven Architecture (MDA) is an architecture for software development whose philosophy is to derive software artifacts from software models [1]. With this architecture, software models are not merely for documentation purpose but are seen as a very high-level programming language. Three main steps are at the core of MDA. First, a software model at a very high level called a platform-independent model (PIM) is created by software designers. PIM concerns only business functionality of the application domain. Second, PIM is transformed into a lower-level software model called platform-specific model (PSM) because this model is tailored for a specific technology platform the will be used to implement the software. Third, PSM is transformed into program code of the chosen platform. The transformation in these three steps is expected to be automatic or semi-automatic with support from software design tools. UML [2] is the software modeling language that is usually associated with MDA.

Business process modeling is receiving much attention in software development community [3]. A business process model can well represent the operational process of a particular business, is easy for the users in the business to understand, and is convenient for the business analysts to define business requirements and communicate with software designers who will develop the corresponding software models.

Our previous work [4] has proposed a guideline called BPM2CD as a complement to the MDA concept. The work suggests how to create a UML class diagram at PIM level for a particular business domain from its business process model represented by a BPMN diagram [5]. The guideline borrows the idea of object-oriented domain analysis [6] to identify UML classes from BPMN processes and adds details to the classes by domain-specific semantics such as software pattern and other additional semantics.

Domain analysis is an important step in the guideline as it discovers key concepts of the application domain from the business process model and proposes them as candidate classes to the software designer. The concept category strategy is one strategy in the domain analysis which involves search in a knowledge base of the domain in order to identify potential classes for the application. This paper sees the benefit of using ontology search here since ontology is a source of conceptual knowledge for a particular domain which itself can link further to ontologies of related domains. With ontology search, we can obtain a refined set of concepts or candidate classes that relate to the particular domain of interest.

We revisit the BPM2CD guideline which is the context of this work in Section II. Section III shows how ontology search is applied to the domain analysis step of the guideline using our search tool. Section IV gives an example using a purchase order domain. Some related work is discussed in Section V and the paper concludes in Section VI.

II. BPM2CD Guideline

The Business Process Model to Class Diagram (BPM2CD) guideline suggests a way a software designer may take to derive a PIM class diagram from a business process model. Fig. 1 shows how the guideline fits in with the MDA concept. The guideline consists of three steps:

W. Rungworawut is with the Information Systems Engineering Laboratory, Department of Computer Engineering, Chulalongkorn University, Bangkok 10330 Thailand (phone: +66 2 2186991; fax: +66 2 2186955; e-mail: wararatkku@yahoo.com).

T. Senivongse is with the Information Systems Engineering Laboratory, Department of Computer Engineering, Chulalongkorn University, Bangkok 10330 Thailand (phone: +66 2 2186996; fax: +66 2 2186955; e-mail: twittie.s@chula.ac.th).

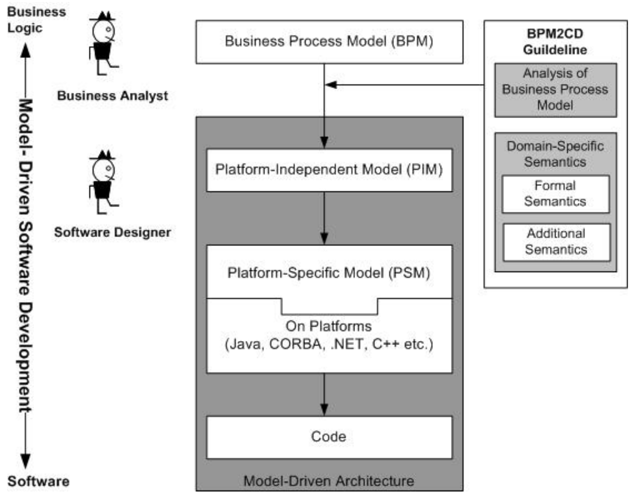

Business Logic
Business Analyst
Software Designer
Software
Model- Driven Software Development
Business Process Model (BPM)
Platform-Independent Model (PIM)
Platform-Specific Model (PSM)
On Platforms
(Java, CORBA, .NET, C++ etc.)
Code
Model-Driven Architecture
BPM2CD Guildeline
Analysis of Business Process Model
Domain-Specific Semantics
Formal Semantics
Additional Semantics

Fig. 1 BPM2CD guideline and MDA

A. Analysis of Business Process Model

In this first step, the business process model is analyzed in order to identify key concepts of the business domain that would constitute a set of classes (i.e., the conceptual model) for the class diagram. This domain analysis approach is an object-oriented analysis technique and comprises two strategies [6]:

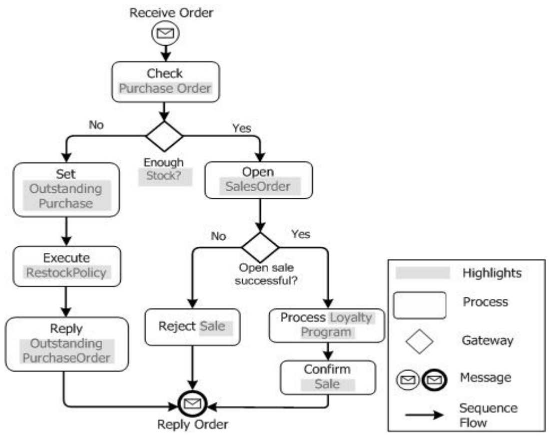
1) Noun Phrase Identification Strategy

The software designer will examine the business process model and identify important noun phrases for the business domain. Fig. 2(a) depicts a business process of a vendor, processing a purchase order, in a BPMN diagram. The purchase order is checked against the stock of goods. If there is enough goods, a sales order is open. If successful, the loyalty program is processed and the sale is confirmed. Otherwise, the sale is rejected. In the case that there is not enough goods in stock, the vendor records an outstanding purchase, executes the restock policy to reorder goods from a supplier, and replies to the customer. The software designer highlights important noun phrases in the business process and these noun phrases become the candidate classes in the conceptual model of the purchase order domain (Fig. 2(b)).

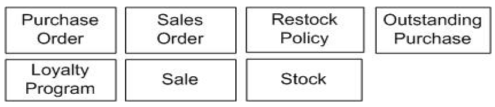
2) Concept Category Strategy

This strategy is based on a collection of vocabularies or concepts that are related to the application domain and are defined by domain experts. The software designer may use noun phrases from the noun phrase identification strategy (e.g., purchase order) to lookup in this collection. Fig. 3(a) shows a table of the concept categories and the corresponding concepts for purchase order. The resulting concepts from the lookup will be chosen as candidate classes by the software designer (Fig. 3(b)).

The two strategies complement each other. The noun phrase identification strategy can discover concepts that are specific to a particular business but may not be listed in the concept category. Likewise, the concept category strategy can discover concepts that are important and should be designed as classes for the application but may be missing from the business process model.

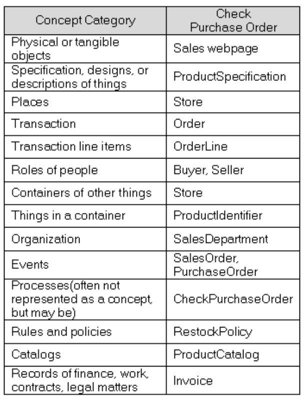


(a) Highlight of noun phrases



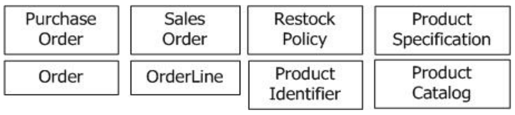
(b) Candidate classes

Fig. 2 Noun phrase identification for purchase order



Concept Category	Check Purchase Order
Physical or tangible objects	Sales webpage
Specification, designs, or descriptions of things	ProductSpecification
Places	Store
Transaction	Order
Transaction line items	OrderLine
Roles of people	Buyer, Seller
Containers of other things	Store
Things in a container	ProductIdentifier
Organization	SalesDepartment
Events	SalesOrder, PurchaseOrder
Processes(often not represented as a concept, but may be)	CheckPurchaseOrder
Rules and policies	RestockPolicy
Catalogs	ProductCatalog
Records of finance, work, contracts, legal matters	Invoice

(a) Concept category



(b) Candidate classes

Fig. 3 Concept category for purchase order

B. Applying Formal Semantics

The software designer may select some concepts from the set of candidate classes from the domain analysis and proposes them as classes for the class diagram. However, these are merely class names. Domain-specific semantics will be applied to add details to the classes (e.g., attributes, methods, relationships between classes) to form the class diagram. Domain-specific semantics exists and may be formal (i.e., properly cataloged) and may be in several forms (e.g., descriptive text, ontology, software model such as software patterns). In [4], archetype patterns related to the domain are applied to the selected classes and form the class diagram.

C. Applying Additional Semantics

Additional semantics refers to other knowledge about the business domain that has not been derived from the domain analysis and may not be cataloged properly. This software designer or business analyst may add additional semantics to refine and complete the class diagram.

III. Ontology Search in Domain Analysis

Ontology is a specification of conceptualization for a domain of interest [8]. It describes knowledge about the domain in terms of concepts or vocabularies within the domain and relationships between them. Several XML-based ontology languages are available (e.g., RDF, RDFS, DAML+OIL, OWL) [9], and they are supported by inference engines. A network of knowledge is achieved by inference and by sharing of ontological concepts among ontologies of different domains.

Looking back at the domain analysis, we see that ontology can help enhance the concept category strategy. As it is a collection of concepts of a domain, the concept category can be represented as an ontology. Therefore we can look for important concepts of the domain from the network of knowledge that it creates.

A. Concept Category Ontology Model

The domain concepts as in Fig. 3(a) can be represented by the concept category ontology model in Fig. 4. This model has three layers. The first layer is the upper ontology. The concept categories (i.e., the left column of Fig. 3(a)) are defined as classes in this ontology. The concepts of the domain (i.e., the right column of Fig. 3(a)) are defined as classes in the lower ontology in the second layer. These classes will be derived from (i.e., be subclasses of) the corresponding categories in the upper ontology. Specific instances of the domain can be defined in the third layer as the instances of the domain concept in the lower ontology.

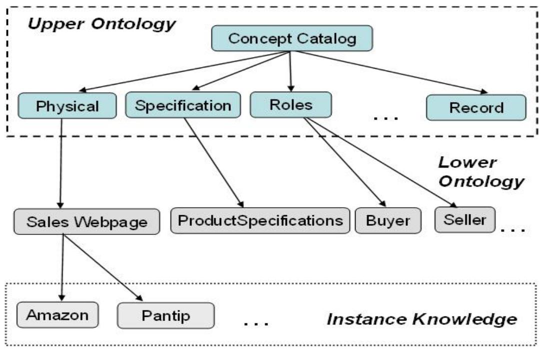


Fig. 4 Concept category ontology model

Fig. 5 shows a snippet of the (lower) ontology of the purchase order domain in OWL [10]. This ontology follows our concept category ontology model and will be used by our ontology search tool to find important concepts for purchase order domain. Note that the domain ontology may not comply with the concept category ontology model; any ontology that defines concepts of the domain as ontology classes can be used with the tool.

```
<?xml version="1.0"?>
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns="http://purchaseOrder.com/purchase.owl#"
   xml:base="http://purchaseOrder.com/purchase.owl">
<owl:Ontology rdf:about=""/>
<owl:Class rdf:ID="SalesWebpage">
   <rdfs:subClassOf>
     <owl:Class rdf:ID="Physical"/>
   </rdfs:subClassOf>
  </owl:Class>
<owl:Class rdf:ID="ProductionSpecification">
   <rdfs:subClassOf>
     <owl:Class rdf:ID="Specification"/>
   </rdfs:subClassOf>
 </owl:Class>
 <owl:Class rdf:ID="Role"/>
 <owl:Class rdf:ID="Buyer">
   <rdfs:subClassOf rdf:resource="#Role"/>
 </owl:Class>
 <owl:Class rdf:ID="Seller">
   <rdfs:subClassOf rdf:resource="#Role"/>
 </owl:Class>
    ....
    ...
</rdf:RDF>
```

Fig. 5 Part of concept category for purchase order in OWL

B. Ontology Search Process

Fig. 6 shows the ontology search process for the concept category strategy. The process begins with the software designer identifying a keyword for the domain. This can be a noun phrase from the noun phrase identification strategy (e.g., purchase order). The keyword is input to our ontology search tool called the concept finder. The tool will look for ontologies that relate to this keyword and extract concepts in the ontologies as candidate classes. The software designer can use the concepts returned from the search as new keywords for lookup.

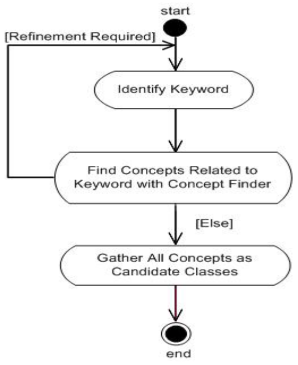

start
[Refinement Required]
Identify Keyword
Find Concepts Related to Keyword with Concept Finder
[Else]
Gather All Concepts as Candidate Classes
end

Fig. 6 Ontology search process

C. *Concept Finder*

Fig. 7 shows the components of the concept finder tool (in dotted box). The software designer inputs a keyword for the domain through the Web-based interface of the tool. The search proxy uses an ontology search engine to retrieve ontology files which contain the keyword. Swoogle [11] is an ontology search engine that performs string search on class names and property names in the ontology files of several languages. Relevant ontologies will be kept in a repository. We assume that domain experts will define concept categories of any domains as ontology files. Another function of the search proxy is to find synonyms for the input keyword. This is because the keyword may not match exactly the term in ontologies. We assume the domain experts also define a dictionary or ontology that specifies synonyms. The synonyms are reported to the software designer.

Relevant ontologies in the repository will be processed by the class extractor component to extract class names. The software designer can use the returned class names or synonyms as the new keywords and repeat the process. This helps refine the search as one keyword will lead to a number of concepts or candidate classes and another keyword will lead to more. The software designer should have a rich set of concepts that cover the knowledge about the domain in a broad area. The software designer selects appropriate concepts which become the classes in the class diagram.

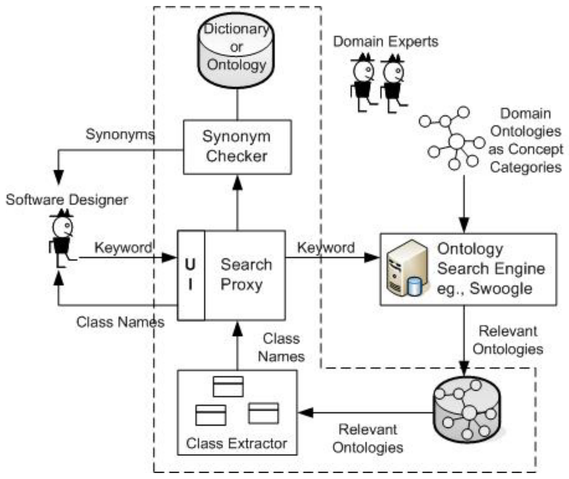


Fig. 7 Diagrammatic overview of concept finder tool

The concept finder tool is implemented using Java Server Page (JSP) [12]. It uses a Jena Java API [13] to read ontology files (RDF, RDFS, and OWL).

IV. Example

This section presents an example of ontology search using the concept finder tool to find candidate classes for the purchase order domain. As in Fig. 8(a), a software designer can input keywords *purchase order* in part 1 of the Web-based user interface. The tool consults Swoogle and returns a list of ontology resources that contain the keywords. It also returns a list of synonyms of *purchase order* (i.e. *order* and *purchase* in this case). In some case when no ontology file that contains the input keywords is found and the tool may discover synonyms of the keyword which can be used instead. The software designer can click on a synonym and it will appear in the keywords box for another search.

Part 2 shows the lists of concepts that are the classes in the ontology resources listed in part 1. The software designer can refine search to obtain more concepts by clicking the radio box in front of the concept that needs to be refined and that concept will appear in the keywords box for search. Suppose that the concept *RestockPolicy* under http://purchaseOrder.com/purchase.owl is clicked and the tool cannot find any ontology files that contain this keyword but its synonym *Reorder* is found. The software designer can search on *Reorder* instead (Fig. 8(b)) and obtains an ontology resource with its concepts listed. In this scenario, the concepts discovered by the keywords *purchase order* and *Reorder* altogether form the set of candidate classes for the purchase order domain.

Using keywords of the domain to search and extract concepts from ontology resources is similar to conventional lookup in the concept category strategy in Section II.A.2 but the tool conveniently allows iterative lookup by other related keywords. When single concept categories may be too general or not complete enough, iterative lookup of related concepts will lead to discovery of more concepts that may not directly belong to the domain but somehow related to it and may be useful for the design of the application. In this way, different sources of knowledge related to the domain are integrated into a more comprehensive one.

Since domain analysis by noun phrase identification strategy and concept category strategy (using the concept finder tool) discovers concepts that are relevant to the domain in general, the analysis result could be a large number of candidate classes, some of which may be or may not be necessary for the specific application being designed. The software designer will have to decide which ones should become classes in the class diagram. Fig. 9 shows the classes that are chosen in this example. As suggested by the BPM2CD guideline, formal and additional semantics can be applied to the chosen classes to give them details and form a complete class diagram. Fig. 10 shows the final class diagram resulting from applying the order and product archetype patterns [7] (above the dotted line) and additional semantics about restock policy and loyalty program (below the dotted line) to the chosen candidate classes.

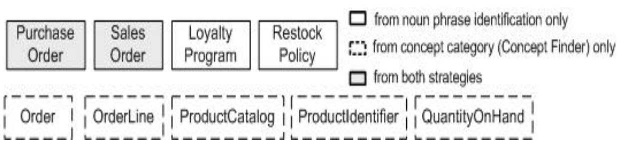
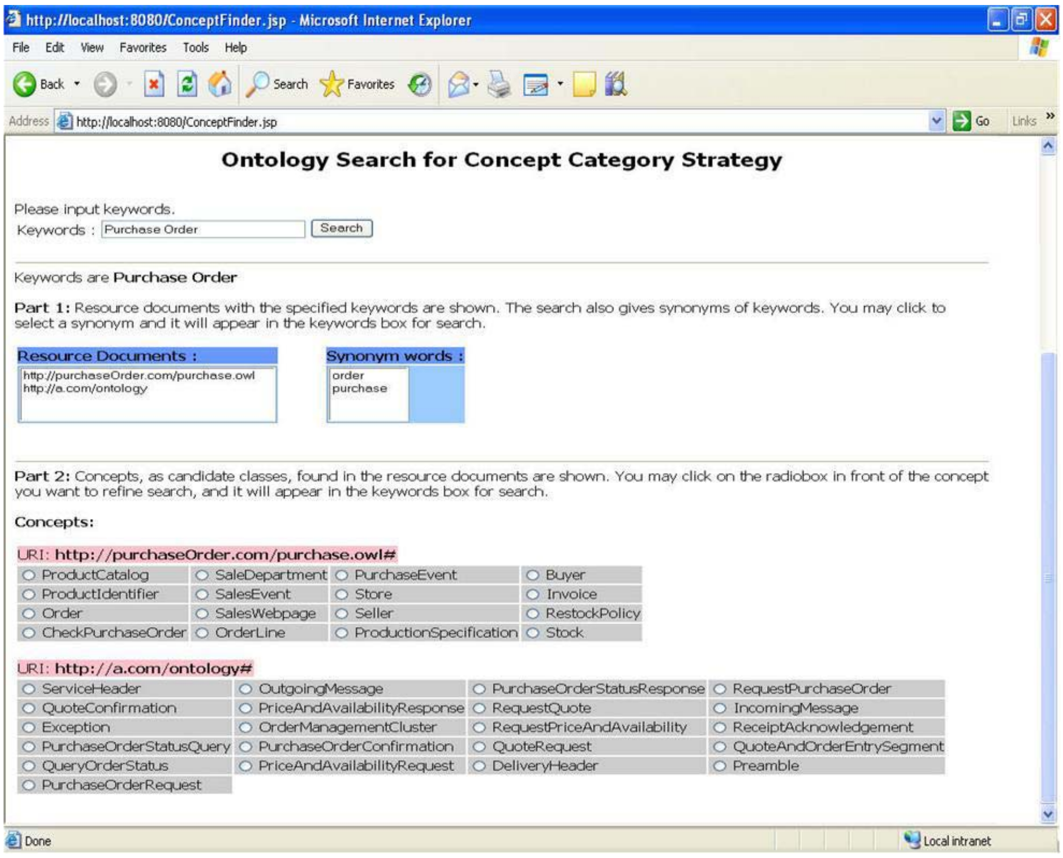


Fig. 9 Candidate classes are chosen



http://localhost:8080/ConceptFinder.jsp - Microsoft Internet Explorer

Address http://localhost:8080/ConceptFinder.jsp

Ontology Search for Concept Category Strategy

Please input keywords.
Keywords : Purchase Order [Search]

Keywords are **Purchase Order**

Part 1: Resource documents with the specified keywords are shown. The search also gives synonyms of keywords. You may click to select a synonym and it will appear in the keywords box for search.

Resource Documents :
http://purchaseOrder.com/purchase.owl
http://a.com/ontology

Synonym words :
order
purchase

Part 2: Concepts, as candidate classes, found in the resource documents are shown. You may click on the radiobox in front of the concept you want to refine search, and it will appear in the keywords box for search.

Concepts:

URI: **http://purchaseOrder.com/purchase.owl#**

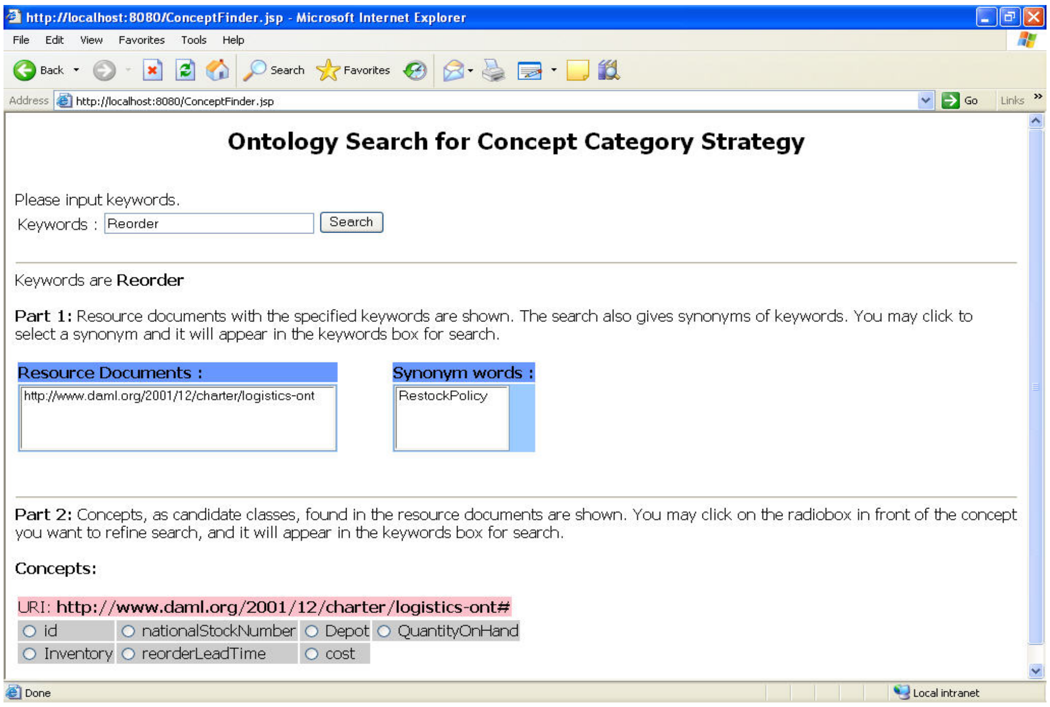
○ ProductCatalog	○ SaleDepartment	○ PurchaseEvent	○ Buyer
○ ProductIdentifier	○ SalesEvent	○ Store	○ Invoice
○ Order	○ SalesWebpage	○ Seller	○ RestockPolicy
○ CheckPurchaseOrder	○ OrderLine	○ ProductionSpecification	○ Stock

URI: **http://a.com/ontology#**

○ ServiceHeader	○ OutgoingMessage	○ PurchaseOrderStatusResponse	○ RequestPurchaseOrder
○ QuoteConfirmation	○ PriceAndAvailabilityResponse	○ RequestQuote	○ IncomingMessage
○ Exception	○ OrderManagementCluster	○ RequestPriceAndAvailability	○ ReceiptAcknowledgement
○ PurchaseOrderStatusQuery	○ PurchaseOrderConfirmation	○ QuoteRequest	○ QuoteAndOrderEntrySegment
○ QueryOrderStatus	○ PriceAndAvailabilityRequest	○ DeliveryHeader	○ Preamble
○ PurchaseOrderRequest			

Done — Local intranet

(a) Search with keyword *purchase order*



http://localhost:8080/ConceptFinder.jsp - Microsoft Internet Explorer

Address http://localhost:8080/ConceptFinder.jsp

Ontology Search for Concept Category Strategy

Please input keywords.
Keywords : Reorder [Search]

Keywords are **Reorder**

Part 1: Resource documents with the specified keywords are shown. The search also gives synonyms of keywords. You may click to select a synonym and it will appear in the keywords box for search.

Resource Documents :
http://www.daml.org/2001/12/charter/logistics-ont

Synonym words :
RestockPolicy

Part 2: Concepts, as candidate classes, found in the resource documents are shown. You may click on the radiobox in front of the concept you want to refine search, and it will appear in the keywords box for search.

Concepts:

URI: **http://www.daml.org/2001/12/charter/logistics-ont#**

○ id	○ nationalStockNumber	○ Depot	○ QuantityOnHand
○ Inventory	○ reorderLeadTime	○ cost	

Done — Local intranet

(b) Refine search with the synonym *Reorder* of the concept *RestockPolicy*

Fig. 8 Using concept finder to search for concepts in purchase order domain

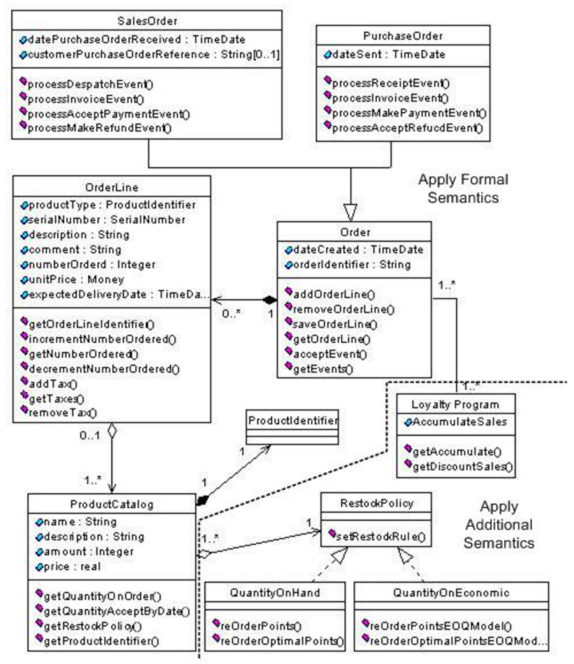


Fig. 10 Appling formal and additional semantics to candidate classes

V. Related Work

Mapping between business process models and UML diagrams has been targeted by a number of researches, and most of the time, it is manual or semi-automatic. The obvious case is the straightforward correspondences between workflow languages and UML activity diagrams such as in [14]. The less obvious case is the mapping to other UML diagrams. In [15], the work focuses on the use of business process patterns and on deriving UML classes from them. Similar to our approach, some semantic information is added to complete the resulting class diagram but no clear guideline has been given on how to identify the classes and where the additional semantics come from. The work provides a supporting tool that helps design business process models and business process patterns, but does not help in mapping to class diagrams.

The idea of our concept finder tool comes from OntoSearch tool [16] which uses Google facility "filetype:RDFs keyword" to search for RDFs ontologies with the specified keyword. The tool can show the ontologies graphically and list all classes in RDFs triple format. Our concept finder instead uses Swoogle which can perform keyword search on several formats of ontology files and the tool is added with the ability to find synonyms for the keywords.

VI. Conclusion

This paper presents a guideline to build a PIM-level class diagram from a business process model of an application domain. The steps to be taken are mostly manual but an ontology search tool is proposed to facilitate the software designers to some extent. The tool assists in the identification of classes for the class diagram by allowing knowledge related to the domain to be discovered.

At present, the concept finder tool can find only 'plain' vocabularies within the domain; it does not distinguish whether a particular concept should be a class name or an attribute name or a method name in the class diagram. We will study on how to make ontology more useful to the building of the class diagrams since correspondences between ontology and UML have been established [17], [18]. The formulation of domain-specific semantics into class diagrams can be more automated such as the formulation from ontology-based domain semantics. Also, the model mapping process could be enhanced for MDA by a formal mapping between the metamodel of the business process modeling language and UML metamodel.

References

[1] A. Kleppe, J. Warmer, and W. Bast, *MDA Explained: The Model Driven Architecture Practice and Promise*. Boston: Addison-Wesley, 2003.

[2] OMG. (2004, October, 2). *UML Specification Version 2.0*. Available: http://www.uml.org

[3] H. Smith. (2003, July). *BPM and MDA: Competitors, Alternatives of Complementary*. White paper. Available: http://www.BPtrends.com

[4] W. Rungworawut and T. Senivongse, "A guildeline to mapping business process to UML class diagrams," *WSEAS Transactions on Computer*, Vol. 4(11), November 2005, pp. 1526-1533.

[5] Business Process Management Initiative. (2004, May, 3). *Business Process Modeling Notation (BPMN) Version 1.0*. Available: http://www.bpmi.org

[6] C. Larman, *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design*. New Jersey: Prentice Hall, Inc., 1997.

[7] J. Arlow and I. Neustadt, *Enterprise Pattern and MDA: Building Better Software with Archetype Patterns and UML*. Boston: Pearson Education, Inc., 2004.

[8] T. Gruber, "A translation approach to portable ontology specifications," *Knowledge Acquisition,* Vol. 5, No. 2, 1993, pp. 199-220.

[9] M. C. Daconta, L. J. Obrst, and K. T. Smith, *The Semantic Web*. Indiana: Wiley, 2003.

[10] W3C. (2004, February, 10). *OWL Web Ontology Language*. Available: http://www.w3.org/TR/owl-features/

[11] umbc.edu. *Swoogle Search and Metadata for the Semantic Web*. Available: http://swoogle.umbc.edu

[12] Sun Developer Network (SDN). *JavaServer Pages Technology : JSP*. Available: http://java.sun.com/products/jsp/

[13] Jena-Semantic Web Framework. Available: http://jena.sourceforge.net/

[14] S. A. White. (2004, March). *Process Modeling Notations and Workflow Patterns*. White Paper. Available: http://www.BPtrends.com

[15] O. H. Barros. (2004, September). *Business Information System Design Based on Process Pattern and Frameworks*. Industrial Engineering Department, University of Chile. Available: http://www.BPtrends.com

[16] Y. Zhang, W. Vasconcelos, and D. Sleeman, "OntoSearch: an ontology search engine," in *Proc. 24th SGAI International Conference on Innovation Techniques and Application of Artificial Intelligence*, Cambridge, UK, 2004.

[17] J. Evermann and Y. Wand, "Toward formalizing domain modeling semantics in language syntax," *IEEE Transaction on Software Engineering*, Vol. 31(1), pp. 21-37, January 2005.

[18] DSTC. (2004) *Ontology Definition MetaModel*, Preliminary Revised Submission to OMG RFP ad/2003-03-40 Volume 1.