Merging Ontologies for Object Oriented Software Engineering

Waralak V. Siricharoen
School of Science, University of the Thai Chamber of Commerce (UTCC)
Vipavadee-Rangsit Rd., Bangkok, Thailand 10700
E-mail: waralak_von@utcc.ac.th, lak_waralak@yahoo.com
http://waralak.mypage.utcc.ac.th

Abstract
Ontologies mentioned in this paper appear in the part of serving the objects identification of object oriented software engineering in initial phase. More ontologies are becoming available on the web, and are increasing in number and demand. Semantic and ontologies search engines have also started to appear, to assist search and recovery of online ontologies which it is the first start to find the ontologies for particular domain of interest. Considering object oriented software engineering, the difficulty of searching the appropriate objects for the system is the one major problem. The main objective of this research is to integrate the different eXtensible Markup Language based (XML-based) formats offered online by using produced Java-based application program to merge and modify ontologies into complete object model. The program will allow the developer to choose objects (extracted from ontologies classes) and their components which are attributes, operations, and relationships. The final result of the program will be used as the sketch object diagrams of the system. in this paper we represent travel domain of interest as the example for the approach.

Keywords: Ontologies, Object Oriented Software engineering, Object Models, Ontology library

1. Introduction

This paper attempts to suggest the view about using what we have to help semi-automatic process of creating object models from ontologies. The similarity between ontologies and object models in object oriented software engineering is the important issue to address in this case. From researcher point of view, the classes in object oriented are similar to classes in ontologies [1]. Many organizations try to construct ontologies for reusing and integrating. So this benefit comes to the interest of supporting software engineering as well. Ontologies are specifications of conceptualizations, used to help programs and humans share knowledge [2]. Ontology provides the fundamental in describing concepts as well as their relationships. It forms the basis in describing how information with a different data structure can be transformed from different application and/or platforms. From a technology point of view, it can be seen as a repository of classes; much like a database represents a repository of data [3]. Ontologies enhance the semantics by providing richer relationships between the terms of concepts/classes. The three major uses of ontologies are: to assist in communication between humans, to achieve interoperability and communication among software systems, to improve the design and the quality of software systems. One significant of the major advantages claimed of ontologies is the potential for the “reuse” of knowledge. We should be able to reuse available ontologies, thereby avoiding the huge effort of starting from scratch [4]. There are many public online ontologies available for reusing. The most of them are stored in ontology libraries. Ontology libraries are storage areas for ontologies. A number of ontology libraries currently exist, hosting various ontology files. In practice they consist of definitions of representational vocabulary, some including axiomatic theories. Most of the files here are in a machine-readable form [4]. There is no one ontology language, therefore ontology libraries are usually restricted to plain text files or some XML-based storage format. In order to achieve an effective level of ontologies reuse, we need semantic search engines capable of helping us find the ontologies we are looking for. Some ontologies search engines have been developed that can provide lists of ontologies that contain specific search terms or keywords, such as Swoogle¹. After all, ontologies are meant to provide an “easy to reuse library of class objects for modeling problems and domains” [5, 6, 7].

This paper indicate the possibilities of using ontologies as the inputs of the semi-automatic object model construction program which will be advantage for object oriented software engineering. The approach is intended to encourage and support reuse of existing ontologies. The program extracts representations of certain classes of retrieved ontologies, and then allows the developers decide on modification object model

¹ http://swoogle.umbc.edu/
components which are attributes, operations and relationships. At the last result, the program will generate the visualize object model as the draft for developers. The similarities between ontologies and object models are addressed in the following section. In this paper presents the selected three XML-based formats of ontologies description from online ontologies which are Resources Description Framework Schema (RDFS), Web Ontology Language (OWL), and DARPA Agent Markup Language (DAML) as the inputs to the program. A full description of the program framework and integrating methodology is given in section 3. An example of how to exploit the program to create object model from ontologies is detailed in section 4 and the last section is the conclusion.

2. The similarities between ontologies and object models

Considering ontology structure also holds definitions of classes, binary relationship between classes and attributes. Relationships may be symmetric, transitive and have an inverse. A minimum and maximum cardinality constraint for relations and attributes may be specified. Concepts and relationships can be arranged in two distinct generalization hierarchies as can see the example of ontologies in Figure 1. in the next page [9].

The object model is the center of object oriented software engineering; on the other hand ontology itself has the class which is the foundation of knowledge base. Ontologies are closely related to modern object-oriented software design [3]. Concepts or classes, which describe concepts in the domain, of ontologies are defined and related to each other, much like classes in an object model [10]. A class in ontologies can have subclasses that represent concepts that are more specific than the superclass as well as in object model. Slots in ontologies are the same as attributes in object models, which describe properties of classes and instances. Concepts, relationship types and attribute abstract from concrete objects or value and thus describe the ontology on the other hand concrete objects populate the concepts, concrete values instantiate the attributes of these objects and concrete relationship instantiate relationships [1].

We can view ontology design as an extension of logical database design, which mean that the guidance object data modelers could be a promise approach. An ontology use the equivalent of database schema but ontology represent a much more richer information model than normal database schema, and also a richer information model compared to class/object model. It is one of the best known modeling languages for real-world projects. The Object Management Group (OMG) has recently issued a call for proposals for a XML-based ontology language which will boost interest in ontology design among software developers. In order to provide some interoperability between ontologies and tools, the plugin has been developed in 2003 [11]. Since then, it has been adopted into routine use by many users. The conversions exist: classes can be compared to ontology classes; objects are similar to ontology instances. attributes and relationships are comparable to ontology slots.

It is suitable to attempt to highlight similarities as well as significant differences in the approach. Ontology are meant to describe and explain the world, while object model(database) are meant to describe that part of the world whose representation has to be managed for some application purpose. The bottom line is that ontologies methodologies show promise as a basis for object oriented software development methodologies.

Figure 1. The example of ontologies library showing in class hierarchy and represent in ontologies description
3. Framework Methodology and Algorithm

3.1 Methodology

The main framework methodology is to use existing ontologies in different format which are DAML, OWL, and RDFS/N3 to discover the objects from ontologies description, and to develop the application program or agent to extract object by integrating online ontologies, and also to use according program to identifying the object and creating object models. In Figure 2. shows the visualized architecture of this system. The processes required by this system are briefly described below.

In Figure 2. shows the whole idea of framework which is started by searching for the ontologies available online. After we got the relevant ontologies, which will usually, come in a different XML-based format language, and we choose them in accordance with their relevance of terms and concepts. The different dot-circle appeared in Figure 2. represents the different ontologies existing online which also have relationships to each other. As our purpose is to use only three well known format of ontologies description which are DAML, OWL, and RDFS/N3. So we select few ontologies we need, and uploaded them as a text files into the program. Then the program will extract the classes/objects and their components. At that moment program will have everything we require in order to create object model. It then merges all objects and developers will come in the part of editing objects and their components as they decide on. In this phase, we use the program as the filter of objects component which some of objects and their components will be drawn off from the object model according to developer's decision.

In this paper we try to build semi-automatic object model by comparing the concepts in the ontology as objects, slot or properties as attributes, and some properties can act as functions or operation. Data (attributes) and functions (methods, operations) are the two fundamental parts of any object. We can continue in this way relying on the decision making abilities of the developer to the final acceptable object. [10]

---

3.2 Merging Algorithm

Next section is clarification the rationale of how to compare the components of ontologies and convert them to the object model. There are three difference formats of ontologies description as mentioned before which are DAML, OWL and RDFS/N3. The classes in ontologies are comparing as classes in object model while the attributes of the ontologies classes are generated as DatatypeProperty. The Generalization/Specialization are mapped as either subClassOf or subPropertyOf depending on the stereotype used. Classes without the specification of any stereotype, represented, would generate a subClassOf. Associations or relationships in object model are translated as ObjectProperty. However, ObjectProperty items are first-class objects while associations-in are not. ObjectProperty can exist independently of classes. Associations-in are defined in terms of association ends, which are related to classifiers. While supports multiple associations, only binary associations are supported in ontologies. Binary associations have only 1 domain class and 1 range class, whereas ontologies property can have multiple domains. It is the direction of the association which determines the source and the object of ObjectProperty. The association ends are translated as Restrictions in ontologies. The property's source becomes a subClassOf Restriction. The Restriction has an onProperty value of the ObjectProperty and a to-class value of the object class. The role of the association with a single value in will generate a cardinality of 1 while an association end
with many values would generate a value of minCardinality [12]. The clearest way of modeling a subproperty relationship is to model the associations involved as a association class. The ontologies language constructs for identity (SameClassAs SamePropertyAs and SameIndividualAs can be defined in a similar style, namely by introducing a constraint with the same name [13,14].

4. How to use the program create object model

This part is to explain the situation of the created program. To help explaining how our proposed framework might work, consider that we are object model developers who are in need of system representing the travel domain. The ontologies are to be used for creating a object model to hold information on Accommodation, Room, Ticket, Activity etc. There are many ontologies online that covers various portions of this travel or tourism domain.

4.1 Selecting Ontologies

First step is to find some potentially relevant ontologies to reuse. This may be done by searching for specific keywords as in Swoogle [3]. This process should produce a list of ontology URIs that are potentially relevant to what ontologies we needs. Lets assume that one of the terms that our system wrote was “Accommodation”. There could be many ontologies out there that covers this concept to some extent, that our developers are not aware of. It might speed up their task if some existing representations can be easily gathered and represented to them to accept, modify, or at least learn from.

4.2 Object Model Creating Program

Now in the program will provide us the all objects extracted from ontologies classes which show in Figure 3. The developers will choose which objects they need by clicking in the box. After choosing the appropriate objects for the system, then the program will show all the object’s components which have been shown in tree view diagram including showing their components extracted from ontologies properties as see in Figure 4.
For example developers extend one attributes ("hasbaggageRoom") in Figure 5. The keyword “Added by designer” will be shown in source column later. Finally the program will show the final result as sketch object model or object diagram as example in Figure 6.

5. Conclusions and Recommendations

Object model is the most important starting point of software analysis and design in object oriented software engineering paradigm. Finding the right objects within the domain of interest is a challenging task for software developers. During the analysis phases, the misunderstanding always occurs between customers and developers. The developers start program with selecting objects and their properties by using group brain storming situation mostly, and start from the requirement specification of the system. This is very time consuming process and need much knowledge and experience of developers. Object modeling’s focus on identity and behavior is completely different from the relational model’s focus on information. This paper assumes the identifying objects are difficult task and there is no exact way to do it. The author believe that using ontologies at this phase makes the developer’s work if not easier much more well-founded and systematic. As a consequence they help reducing developing time and increase efficiency and effectiveness of the development process.

An ontology is an defined description of concepts and relations that exist in a particular domain such as a given organization, a study field, an application area, etc. The main properties of ontology are sharing and filtering. Sharing means that an agreement may exist between different agents based on the acceptance of common ontologies, that they have the same understanding of a given concept. Ontologies declare hierarchical categories and valid relationships. When two people are talking about terms or objects, they are talking about the same concept. Filtering is linked to abstraction when we are considering models of reality.

This methodology, by definition, takes into account only a part of the reality. Their usefulness is based on their ability to filter out a lot of undesirable characteristics. Ontology defines what should be extracted from a program in order to build a given model of this system. The definition of ontology here corresponds closely to the classical definition of a meta-model. Generalizations and specializations used in ontologies are the same concepts used in the object-oriented software engineering process.

This paper presented a new approach for semi-automatic construction of proper object model for the system in domain of interest. In this paper, we present the travel/tourism domain as illustrated example. The idea is based on reusing the increasing number of online ontologies to build new object model, rather than the current costly habit of starting from scratch. The proposed system intends to make use of a number of technologies to complete its task, such as ontology searching, merging, and editing. Object models and ontologies have some similarities as well as differences. At the center of both object models and ontologies are objects within a given problem domain. The difference is that while the object model should contain explicitly shown structural dependencies between objects in a system, including their properties, relationships, events and processes, the ontologies are based on related terms only. On the other hand, the object model refers to the collections of concepts used to describe the generic characteristics of objects in object-oriented languages. Ontology is accepted as a formal, explicit specification of a shared conceptualization, we can naturally link ontologies with object models, which represent a system-oriented map of related objects. In this paper has discussed how ontologies and object models can be mapped to each other. However, only the ontologies are not enough. The functions/methods are fundamental elements of the complete object model, consisting of developer-defined concepts (objects) and represented as Object model. Defining those elements require additional
complete ontologies of system functionality (behavior) combining with the intuitive and experience of developers. This paper deals with merging ontologies with existing methods, techniques, and tools used and shows that they can contribute significantly to reaching better decisions, with a positive effect on all the subsequent phases of the development process. This paper describes a framework methodology for supporting the high-level analysis phase of object-oriented software engineering using ontologies for identification of system objects.

As the author recommendation, applying proper ontologies at the first place might help selecting the right objects. However, it cannot be guaranteed that the segments extracted from the original ontologies will retain the quality and consistency of their source. The program proposed in this paper is reusing existing ontologies; the quality of those ontologies will certainly affect the quality of the output ontology. So it comes to the point that developers are the still the significant factor for software engineering approach. Because they might want to limit the system to only those ontologies that pass certain quality, or they want to edit more components in the program to complete the system object models as the results.

6. References


